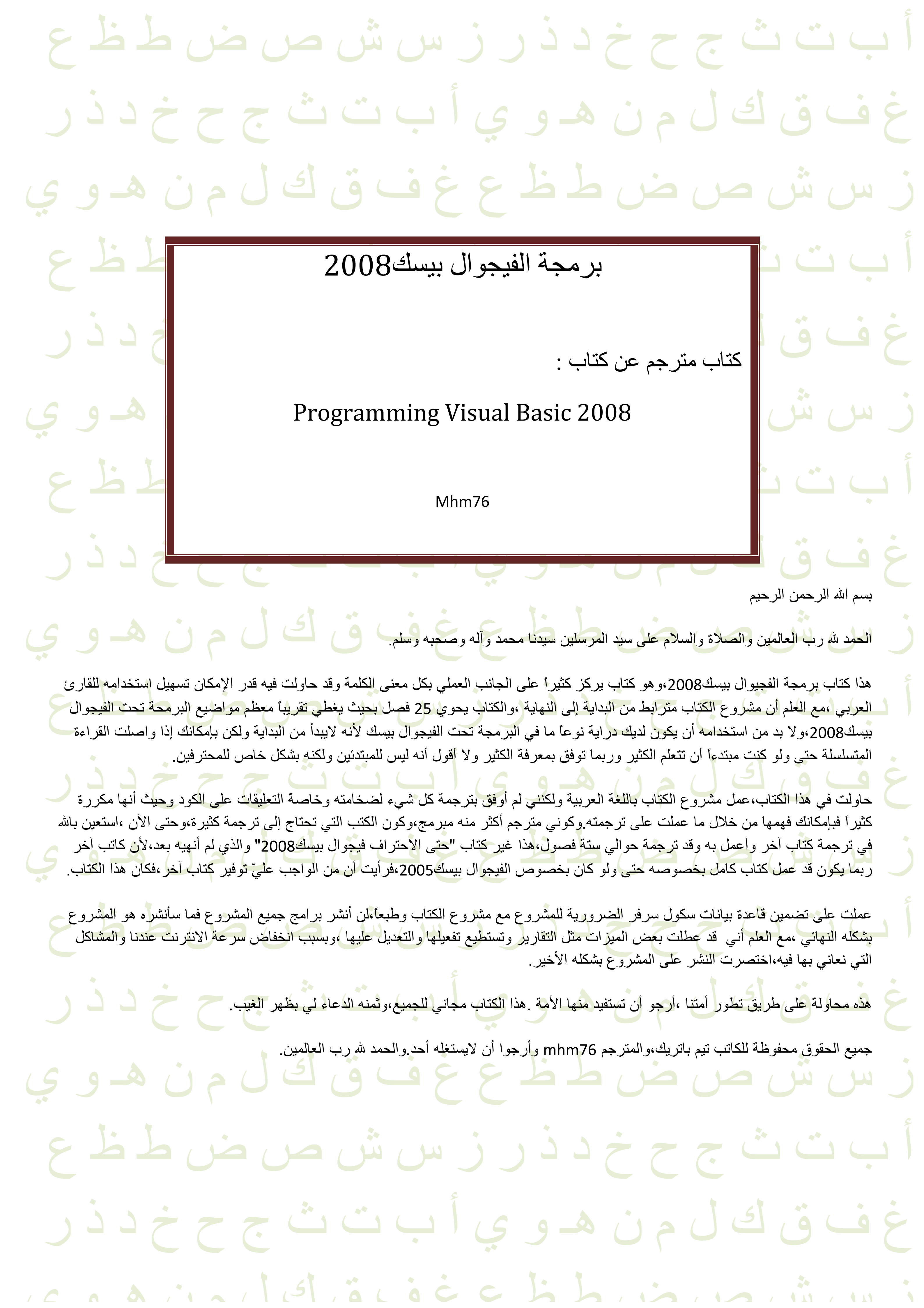


برمجة الفيجوال بيسك2008

كتاب مترجم عن كتاب :

Programming Visual Basic 2008

Mhm76

بسم الله الرحمن الرحيم

الحمد لله رب العالمين والصلاة والسلام على سيد المرسلين سيدنا محمد وآله وصحبه وسلم.

هذا كتاب برمجة الفجيوال بيسك2008،وهو كتاب يركز كثيراً على الجانب العملي بكل معنى الكلمة وقد حاولت فيه قدر الإمكان تسهيل استخدامه للقارئ العربي ،مع العلم أن مشروع الكتاب مترابط من البداية إلى النهاية ،والكتاب يحوي 25 فصل بحيث يغطي تقريباً معظم مواضيع البرمجة تحت الفيجوال بيسك2008،ولا بد من استخدامه أن يكون لديك دراية نوعاً ما في البرمجة تحت الفيجوال بيسك لأنه لايبدأ من البداية ولكن بإمكانك إذا واصلت القراءة المتسلسلة حتى ولو كنت مبتدءاً أن تتعلم الكثير وربما توفق بمعرفة الكثير ولا أقول أنه ليس للمبتدئين ولكنه بشكل خاص للمحترفين.

حاولت في هذا الكتاب،عمل مشروع الكتاب باللغة العربية ولكنني لم أوفق بترجمة كل شيء لضخامته وخاصة التعليقات على الكود وحيث أنها مكررة كثيراً فبإمكانك فهمها من خلال ما عملت على ترجمته.وكوني مترجم أكثر منه مبرمج،وكون الكتب التي تحتاج إلى ترجمة كثيرة،وحتى الآن ،استعين بالله في ترجمة كتاب آخر وأعمل به وقد ترجمة حوالي ستة فصول،هذا غير كتاب "حتى الاحتراف فيجوال بيسك2008" والذي لم أنهيه بعد،لأن كاتب آخر ربما يكون قد عمل كتاب كامل بخصوصه حتى ولو كان بخصوص الفيجوال بيسك2005،فرأيت أن من الواجب عليّ توفير كتاب آخر،فكان هذا الكتاب.

عملت على تضمين قاعدة بيانات سكول سرفر الضرورية للمشروع مع مشروع الكتاب وطبعاً،لن أنشر برامج جميع المشروع فما سأنشره هو المشروع بشكله النهائي ،مع العلم أني قد عطلت بعض الميزات مثل التقارير وتستطيع تفعيلها والتعديل عليها ،وبسبب انخفاض سرعة الانترنت عندنا والمشاكل التي نعاني بها فيه،اختصرت النشر على المشروع بشكله الأخير.

هذه محاولة على طريق تطور أمتنا ،أرجو أن تستفيد منها الأمة .هذا الكتاب مجاني للجميع،وثمنه الدعاء لي بظهر الغيب.

كائنات الدوت نت The .NET Object.

من أجل الفهم الكامل لتطوير البرمجيات في الدوت نت،يجب أن تفهم ما هي الكائنات.(إذا كنت على دراية بالبرمجة الموجهة بالكائنات ،بإمكانك الذهاب إلى المقطع التالي،ولكن ستفقد بعض من المحتوى الهام).وحيث أن بعض من هذه المعلومات في هذا المقطع ستظهر في الفصل الثامن،يبقى من الهام شرح بعض من مفاهيم الدوت نت هنا.

الكائنات والبيانات Objects and Data.

من وجهة نظر البرمجة،يقوم الكمبيوتر بعمل أربع مهام أساسية:

- ✓ إنه يخزن البيانات dataفي المنطقة الخاصة بذاكرة الكمبيوتر.
- ✓ إنه يدعم معالجة هذه البيانات data من خلال العمليات الأساسية المتضمنة الجمع ،الطرح،والجبر المنطقي،ومعالجة المتغيرات النصية للنصوص.
- ✓ إنه يسمح للمستخدم من التفاعل مع البيانات data المخزنة في الذاكرة.
- ✓ إنه يوفر طريقة ما لجلب البيانات data وإدخالها بالإضافة إلى إخراجها من الذاكرة،من خلال وسائل الإدخال والإخراج مثل لوحة المفاتيح(إدخال) والطابعات(وسائل إخراج)وكذلك وسائل التخزين المختلفة مثل السواقات.

إن جوهر هذه المهام الأربعة هو البيانات data. يوجد الكمبيوتر لمعالجة البيانات، توفر أنظمة التشغيل حجر الأساس لهذه المهام .ولكنها تطبيقات برمجية تنشئ القدرة لهذه الميزات features—the ability لكي تعالج بيانات حقيقية data—real ذات معنى للمستخدم.إن لغات البرمجة العالية المستوى High-level programming languages هي الأدوات الأساسية المستخدمة لتطوير هذه التطبيقات، وتستخدم كل منها بعض الطرق العامة لجعل ميزات معالجة البيانات متاحة للمبرمج.

بالعودة إلى الأيام الذهبية للبرمجة بلغة التجميع assembly، و إذا علمت قسم عنونة الذاكرة memory address لقطعة من البيانات،فبإمكانك الوصول إليها ومعالجتها مباشرةً. في الأيام الأولى للغة البيسك وفي أغلب لغات البرمجة(الإجرائيةprocedural)الأخرى،كان الوصول للبيانات يتم من خلال المتغيرات*variables* . وبما أن اللغات قد تطورت في التعقيد والأهداف،وعليه فقد تطور عرضها للبيانات.فيما يخص لغةLISP (اختصار ل List Processing أو Lots of Irritating Silly Parentheses)،فإنه تتواجد أي قيمة للبيانات ضمن قائمة list أو مجموعة set كبيرة من البيانات.لكن في لغات الدوت نت،فإنه يتم عرض البيانات من خلال الكائنات object .

فالكائنات هي مجموعة قيم البيانات ومصحوبة بالكود المصدري. بينما في لغات البيسك القديمة، كان كل عنصر من البيانات أكثر أو أقل استقلالية من خلال متغيره المحجوز(أو المخصص)، المرتبط بقيم البيانات في اللغات الكائنية التوجه(أو الموجهة بالكائناتOOP languages)التي يمكن تصنيفها إلى كائنات. وغالباً ما تتضمن تنفيذات الكائن كود مصدري مصمم لمعالجة قيم البيانات لذلك الكائن.

تمثل الكائنات بشكل عام شيء ما، وغالباً ما يكون لهذا الشيء مثيل(نظير)في العالم الحقيقيreal-world،إما مادياً أو ظاهرياً(معنوياً).على سبيل المثال،من المحتمل أن يتضمن كودك كائن المنزل الذي له حقول fields بيانات أو خصائص properties للعنوان، كـــلون الدهان الخارجي ، وعدد الناس القاطنين في المنزل. فالكود المصدري المساعد يمكنه أن يدير تلك البيانات: فطريقة method الطلاء تغير قيمة اللون المستخدمة للدهان الخارجي. تدعى البيانات وعناصر الكود ضمن كائن ما بالأعضاء members.وإن بعض الأعضاء مخفي داخل الكائن ويتم الوصول إليها فقط بالكود المصدري للكائن. الأعضاء الأخرى أكثر عمومية :حيث يمكن لأي كود في تطبيقك أن يستخدمها ،وهي ليست مجرد مجموعة جزئية من كود تطبيق موجود داخل الكائن. اعتبر التلفاز ككائن ما، فالأعضاء العامة للتلفاز على العموم هي سهلة الاستخدام : زر الطاقة ،جهاز التحكم ،التحكم بالصوت ،وهكذا . وهذه المسالك التي من خلالها يتحكم المستخدم بقيم بيانات التلفاز (مخرجات الفيديو والايديو).توجد أيضا أعضاء مخفية داخل التلفاز : بإمكانك استخدام هذه الأعضاء لتؤثر على الصورة وجودة الصوت ،على الرغم من أن هذا سيكون فكرة سيئة لأغلب المستخدمين .وبنفس الطريقة لايحتاج الكائن لكود خارجه للعبث بمكوناته الداخلية في ماعدا الأعضاء العامة . لاأهتم كيف يعمل التلفاز داخلياً ، طالما أنني أستطيع الحصول على الصورة والصوت الخارج منه باستخدام المتحكمات (الأدوات) التي تم عرضها(الطاقة ، القناة ، الصوت).

الكائنات والواجهات Objects and Interfaces

تمثل المكونات العامة للكائن واجهته *interface*.فإذا مأأراد كود من خارج الكائن معالجة البيانات المنتمية لذلك الكائن، فإنه يستخدم أعضاء الواجهة. ليس عليه أن يفهم الأعضاء المخفية أو كيفية عملها،وذلك جيد . وخاصة إذا تغيرت هذه الأعضاء الداخلية لأي سبب كان، من الممكن أن يحدث غالباً أكثر مما تعتقد. تعمل الكائنات في البرمجة الموجهة بالكائنات بنفس الطريقة (طريقة تطور التلفاز أنه كان يملك واجهة كبيرة ويشغل حيزاً واسعاً أما الآن فإنه يملك أدوات تحكم عن بعد ويشغل حيز أقل ويحتوي على قنوات ووظائف أكثر ولكن الواجهة تقريباً نفسها). طالما أن الواجهة العامة بقيت تقريباً كما هي ، فإن الكود الفعلي للكائن ونظام تخزين بيانات داخلية معروف أيضاً بتنفيذ الكائن يمكن تغييره دون التأثير على كامل التطبيق.

الكائنات والحالات(النسخ أو الصور عنها)Objects and Instances

بالحقيقة إن الواجهة و التنفيذ لكائن تمثلان هيكله فقط ،وهذه هي الأجزاء التي ينشئها المبرمج من خلال الكود المصدري . وحتى أنها توجد قبل أن ينصب ويترجم البرنامج على كمبيوتر المستخدم . في الحقيقة ، في هذا المستوى ، حتى أن الكائنات لا تُعرف بشكل حقيقي بالكائنات . في معظم اللغات (ومن ضمنها الفيجوال بيسك) ، فإن الكلمة فئة class تشير إلى عملية تنفيذ واجهة كائن ما. حالما يتم تنصيب تطبيقك على الكمبيوتر وتشغيله ، فإن الكود ينشئ نسخة instances من الفئة ليخزن البيانات الفعلية في الذاكرة . هذه النسخ (الصور) هي الكائنات الحقيقية من تطوير البرمجة الموجهة بالكائنات OOP. اعتماداً على طريقة كتابة كودك ، فربما يتم استخدام تنفيذ فئة وحيدة لإحداث واحدة أو حتى المئات من الكائنات في الذاكرة بنفس الوقت.

في الدوت نت، تظهر جميع قيم كودك وبياناتك داخل الكائنات . وجميل جداً أن يكون كل شيء تراه في برنامج التنفيذ هو كائن ما :حيث أن نموذج ويندوز هو كائن ما : أداة صندوق القائمة list box التي على ذلك النموذج هي كائن ما : وبند مفرد في صندوق القائمة ذاك هو كائن ما.

أجزاء إطار عمل الدوت نت The Parts of the .NET Framework

يوجد هناك الكثير من أجزاء إطار عمل الدوت نت التي ما زالت قيد المناقشة . وتظهر هذه الأجزاء كثيراً في توثيق الدوت نت ،وكل منها لديه كلمة مركبة من ثلاثة حروف three-letter acronym (ALT) ، أو ما شابه ذلك .

لغة التنفيذ المشتركة The Common Language Runtime

تقع لغة التنفيذ المشتركة(CLR) *Common Language Runtime* موقع القلب بالنسبة لإطار عمل الدوت نت ، فكل شيء تعمله في برنامج الدوت نت تتم إدارته *managed* بوساطة لغة التنفيذ المشتركة CLR . عندما تنشئ متغيراً ما ، اشكر لغة التنفيذ المشتركة CLR ونظام إدارة بياناتها

management.عندما تعمل على حذف قطعة من البيانات ، اشكر لغة التنفيذ المشتركة CLR وكيف تدير manage التخلص من البيانات من خلال نظام تجميع النفايات التابع لها. هل لاحظت كم يتكرر ظهور الكلمة manage في تلك الجمل ؟ ولكن التعبير المناسب ، هو ماذا تعمل لغة التنفيذ المشتركة . في الحقيقة ، تدعى البرامج المكتوبة من أجل إطار عمل الدوت نت الكود المدار managed code. أي كود يحدث خارج سيطرة لغة التنفيذ المشتركة ، المتضمنة مكونات COM (ActiveX) المستعملة من قبل تطبيقك الدوت نت ، المعروف بالكود الغير مدار unmanaged code . تشبه لغة التنفيذ المشتركة كثيراً مطار لوس انجلوس الدولي . فالطائرات تصل و تغادر في كل دقيقة . السيارات بالآلاف تدخل وتغادر طريق الأوتوستراد و مركز المباني العامة . يتحرك الناس واللصوص باستمرار بين المحطات الرئيسية الثمانية والمحطة الدولية الضخمة . يوجد أحداث كثيرة ، ولكن تتم إدارة الكثير منها. فالطائرات لا يمكنها الإقلاع أو الهبوط بدون موافقة على ضبط التحليق . تدير البوابات و نقاط الوصول الطرق و الكراجات العامة . بصدق ، تدير وكالات TSA الجيدة طيران المسافرين و اللصوص داخل وخارج المناطق المأمونة للمحطات . تؤكد هياكل الإدارة والضبط في مكان ما في LAX بشكل مرتب ومضمون طيران الناس بين طائراتهم ومدنهم من لوس انجلوس . وتؤكد هياكل الضبط والإدارة في لغة التنفيذ المشتركة CLR بشكل مرتب ومضمون على تدفق البيانات بين كود الدوت نت وبقية الكمبيوتر أو الشبكة المتصلة .و من المحتمل أنك تود أن تعرف عن سر كيفية كون لغة التنفيذ المشتركة قادرة على معالجة البرامج المكتوبة في أي لغة من لغات الدوت نت ، المتضمنة الفيجوال بيسك و سي شارب والفورتران . وهذه هي إرادة المتنافسين في الميكروسوفت . بالواقع إنهم يعرفونه بالفعل، لأنه لا يوجد سر.تحول كل لغات الدوت نت الممكنة (بمعنى تترجم) كودك المصدري إلى اللغة ميكروسوفت الوسيطة*Microsoft Intermediate Language*(أو*MSIL* ، وبشكل عام اختصرت *IL*) وهي اللغات المعروفة من قبل لغة التجميع assembly language التي تبدو غير مفهومة . على سبيل المثال ، يوجد هنا بعض الكود المصدري للفيجوال بيسك لتطبيق الكونسول الذي يطبع بشكل مبسط "Hello, World!"، من إجرائية الكود المسماة Main :

```
Module Module1
    Sub Main()
        Console.WriteLine("hell world")
        Console.Read()
    End Sub
End Module
```

وهو برنامج الدوت نت الكامل .عندما يحوله مترجم الفيجوال بيسك إلى لغة الميكروسوفت الوسيطة MSIL ، تبدو الإجرائية Main مشابهة لهذه (نوعاً ما معدلة لتتناسب مع هذه الصفحة)

```
.method public static void Main( ) cil managed
{
        .entrypoint()
.custom instance void [mscorlib]System.
STAThreadAttribute::.ctor( ) = ( 01 00 00 00 )
// Code size 11 (0xb)
        .maxstack(8)
IL_0000: ldstr("Hello, World!")
IL_0005: call
void [mscorlib]System.Console::WriteLine(string)
IL_000a: ret()
} // end of method Module1::Main
```

نعم،إنه مبهم.ولكنه مناسب، تفهمه لغة التنفيذ المشتركة CLR ، وهذا ما نعتبره فعلاً في الدوت نت . طالما أنك تستطيع أن تحصل على كودك داخل اللغة الوسيطة IL ،فإن الدوت نت سوف تعالجه. يحدث مترجم الفيجوال بيسك تماماً ليولد اللغة الوسيطة IL من أجلك . و كذلك مترجمات لغة الدوت نت الأخرى، المتضمنة سي شارب ، واللغة الوسيطة IL الهدف. حتى أنه يمكنك أيضاً أن تكتب كود اللغة الوسيطة IL الخاص بك، لكن من المحتمل أنك تقرأ الكتاب الخطأ من أجل ذلك . فقط طمئن نفسك ، سيكون هذا الجزء الأخير من اللغة الوسيطة IL التي تراها في هذا الكتاب .

توصيف اللغة المشتركة The Common Language Specification

اللغات التي تدعي دعم الدوت نت لا يمكنها فقط أن تقول ذلك لأي سبب قديم . في الواقع عليها أن تكون منسجمة مع الدوت نت و أعمالها. و قد تم عمل ذلك من خلال توصيف اللغة المشتركة Common Language Specification (CLS) . يعرف توصيف اللغة المشتركة CLS أصغر مجموعة من الميزات التي يجب على اللغة أن تنجزها قبل أن تعتبر خاضعة للدوت نت ، أو بدقة ،خاضعة لتوصيف اللغة المشتركة CLS-compliant . يمكن أن تذهب اللغة إلى ما وراء الحد الأصغر إذا أرادت ذلك ، و تتضمن الدوت نت العديد من الميزات الإضافية فوق ميزات اللغة المحددة التي يمكن أن تبنى . من الممكن أن لا تكون اللغة التي تنفذ الحد الأصغر فقط لتوصيف اللغة المشتركة المحددة قادرة على التفاعل الكامل بين مكونات اللغات التي تجاوزت توصيف الحد الأصغر. الفيجوال بيسك هي ، بالطبع ، تابعة لتوصيف اللغة المشتركة ، و بالحقيقة إنها تبتعد خلف الحد الأصغر.

نظام الأنواع المشتركة The Common Type System

بما أن لغة التنفيذ المشتركة CLR تتحكم بكودك المصدري بأي حال ، فإن الميكروسوفت اعتقدت أنه من الجيد أن تملك السيطرة على بيانات الكود المصدري أيضاً . يعمل إطار عمل الدوت نت هذا من خلال نظام الأنواع المشتركة Common Type System (CTS)، الذي يعرف أنواع البيانات الأساسية وآلية البيانات المستخدمة في برامج الدوت نت. وهو يتضمن جميع أنواع القيم: العددية، السلاسل النصية والأنواع المنطقية . وهو يعرف أيضاً الكائن object،وهو وحدة تخزين البيانات الجوهرية في الدوت نت .

يقسم نظام الأنواع المشتركة CTS جميع الكائنات إلى مقدارين كبيرين . يدعى المقدار الأول: الأنواع ذات القيمة value types،ويخزن البيانات الفعلية الصحيحة في الخانات . إذا كان لديك قيمة عددية صحيحة 32 integer بت32-bit، فإنها تدخل التعديل في خانة نوع القيمة ، الجاهزة لاستخدامك المباشر. ويحتوي المقدار الآخر: الأنواع المرجعية reference types. عندما تنظر في هذا المقدار، فإنك ترى خريطة تخبرك أين تجد البيانات الفعلية

بطريقة ما في ذاكرة الكومبيوتر. ومن الواضح أن الأنواع ذات القيمة هي المرجحة وهي أبسط للاستخدام ، وهي كذلك، ولكن تصاحبها العديد من القيود،التي لا تُفرض على الأنواع المرجعية.
يمكن للمكونات و البرامج المكتوبة باستخدام نظام الأنواع المشتركة القياسي أن تتبادل البيانات مع بعضها البعض بدون أي عوائق hindrances أو قيود. (يسقط القليل من أنواع بيانات الدوت نت خارج الجوهر"core" لأنواع نظام الأنواع المشتركة ، لكن عليك أن تتجنبها فقط و خاصة عندما تريد أن تتفاعل مع المكونات التي يمكن أن تستخدم جوهر الأنواع لنظام الأنواع المشتركة فقط.) عندما تكتب تطبيقاتك في الفيجوال بيسك ، فإن معظم كودك سيظهر في فئات.classes. والفئات هي أنواع مرجعية وهي تتضمن كلاً من قيم البيانات والكود المصاحب .إن قيم البيانات الموجودة في فئة ما معظمها على الأغلب الجوهر لأنواع بيانات نظام الأنواع المشتركة ، لكن يمكن أن تحتوي أيضاً على الكائنات التي تصممها في مكان آخر في تطبيقاتك .وتتضمن الفيجوال بيسك أيضاً التراكيب,structures،و أنواع قيمة تنفيذ التراكيب، وتتضمن أيضاً كل من البيانات و الكود.
الفئات و التراكيب هي تماماً نوعان من البيانات / الكود المتوفرة في الفيجوال بيسك . الواجهات هي فئة وهياكل تركيب structure skeletons وتتضمن الواجهات تفاصيل تصميمية لما يجب أن يظهر في تركيب أو فئة مترابطة ، ولكن لا تتضمن أي تنفيذ فعلي أو كود شغال. تعرف التفاويض إجراء ما بدون تنفيذ (توقيع "signature")، ويتم استخدامه لدعم الأحداث events ،هذه الأعمال(الممهد لها من قبل المستخدم ، ونظام التشغيل ، أو كودك) التي تخبر كودك "Get to work now!". الوحدات البرمجية Modulesهي مقاطع من الكود والبيانات ، ولكن لا تشبه الفئات والتراكيب ، وإنه ليس بإمكانك أن تنشئ كائنات مستقلة منها . مجموعة العدادات Enumerationsهي مجموعة مرتبطة بالقيم الصحيحة ، و هي عموماً من أجل الاستخدام كقائمة من الاختيارات .
في لغة الدوت نت ،إن جميع هذه المصطلحات (فئة class ، تركيب structure ، واجهة interface ،تفويض delegate ، وحدة برمجية module ، العداد enumeration) تعرف إجمالاً بالأنواع .types. من المحتمل أنك عرفت للتو أن للدوت نت عناصر مشوشة فيها: لذلك لن تشتري كتاب يتعلق بها إن كانت سهلة. ولكن على الرغم أن جميع التقنيات معقدة ، فإن الكلمة البسيطة، نوع type، هي التي تسبب الإرباك الأكبر .إنك ستلاقي بعض الخوف طيلة هذا الكتاب عند قراءته كل مرة . المشكلة : ليست فقط أن النوع يشير إلى تلك العناصر الجوهرية لنظام الأنواع المشتركة CTS لكن يتم استخدامه عند التحدث عن أنواع القيمة الخاصة بالفيجوال بيسك (تدعى أكثر الأحيان الفيجوال بيسك بأنواع البيانات "data types") . واللقب للتراكيب هو أنواع معرفة بالمستخدم "user-defined types," ومع ذلك يوجد التباس آخر لاستخدام النوع ".type". يذكر المبرمجون الذين استخدموا الفيجوال بيسك قبل تجسيده للدوت نت أيضاً أن النوع "Type" كعبارة لغوية اعتادت أن تنشئ أنواع معرفة بالمستخدم user-defined types . على الميكروسوفت أن تستخدم بعض الكلمات الأخرى للعالم بدلاً من نوع type كالفئات classes ، الواجهاتinterfaces ، العدادات enumerations ، وهكذا . سيكون اختيار أي كلمة أفضل بما أنه سيتم استخدامها أحياناً لتناقش البرمجيات . لكن "type"هو الكلمة المختارة، لذلك من الأفضل لك أن تعتاد على رؤيتها .
تتألف عادة أعضاء النوع من حقول بيانات بسيطة و إجرائيات كود ، لكن يمكنك أن تدخل أيضاً أنواع أخرى كأعضاء . حيث أنه ، يمكن لفئة ما أن تتضمن فئة ما متداخلة nested إذا دعت الحاجة لذلك . تدعم أشكال محددة فقط التداخل . أتحدث أيضاً عن مستويات الوصول access levels في هذا القسم . كل عضو يملك مستوى دخول و الذي يقول ما الكود الذي يستطيع استخدامه ذلكٍ العضو . يوجد خمسة مستويات دخول access ،تتراوح من العام (شخص ما و إخوته يمكن أن يستخدموا الأعضاء) إلى الخاص private (عليك أن تدخل النوع كي تعرف أنه موجود هناك) . الفصل السادس يناقش نظام الدوت نت بـأجل التفاصيل ، المتضمنة المعلومات التي تحتاجها عن الفئات ، والتراكيب ، إلخ.

مكتبات فئات الدوت نت .NET Class Libraries

إن الكمبيوترات بدقة مبهمة تماماً . في حين أنني أستطيع العد بجميع الطرق حتى العدد 17 ، أما الكومبيوتر فأعظم قيمة للعد هي العدد 1 : إنه يعرف فقط الأرقام 0 و1 . تتضمن وحدة المعالجة المركزية CPU مجموعة من العمليات البسيطة المستخدمة لمعالجة الأرقام 0 و1، وعمليات أكثر قليلاً تقارن بين 1sو0s بطرق معقدة . والحيلة العظيمة الأخيرة للكمبيوتر هي قدرته على نقل 0sو1s من و إلى الذاكرة . تأكد أنه يعمل هذه الأشياء بسرعة الضوء تقريباً ، ولكن هل يستطيع أن يحسب 3.14 حتى 3 مليون خانة عشرية ؟ حسناً بالفعل إنه يستطيع. لا تعرف الكمبيوترات أي شيء عن حروف الأبجدية ، وهي بحق يمكنها معالجة الأرقام 0 و 1 فقط ، ومع ذلك أستطيع استخدام الكمبيوتر هنا لكتابة كتاب ربح جائزة award-winning. إنها القدرة على تصنيف قيم البيانات البسيطة 1 بت و المعاملات داخل مكتبات libraries معقدة بشكل متزايد للتخصص الوظيفي التي تجعل فائدة الكومبيوتر ممكنة.
يبنى عمل إطار الدوت نت على عشرات من التخصص الوظيفي المعقد بشكل متزايد. عندما تنصب إطار عمل الدوت نت ، فإن وحدة المعالجة المركزية ونظام النوع المصاحب لها يمثلان الجوهر لإطار العمل . يتضمن إطار العمل بحد ذاته جميع التخصص الوظيفي الأساسي الذي يحتاجه لتتمكن من أن تضيف 2 إلى 2 وتحصل بشكل صحيح على 4 . وبصفتك مطور تطبيق عمل ما، ستقضي وقتاً طويلاً لعمل ذلك فقط. لكن ماذا لو أردت أن تعمل شيء ما أكثر تعقيداً ، شيء ما والذي تعرف بعض البرامج التي نفذته ، كتصنيف قائمة من الأسماء أو رسم دائرة ملونة على نموذج ما؟ لتحصل على الجواب ، اذهب إلى class libraries،مكتبات فئات الدوت نت . تتضمن هذه المكتبات ، المنصبة مع إطار العمل ، الكثير من (المعقدة بشكل متزايد)التخصص الوظيفي المكتوب سابقاً والذي ليس عليك أن تكتبه من الخربشة.
يوجد مكتبتا فئة في الدوت نت : مكتبة الفئة القاعدية Base Class Library (BCL) ومكتبة فئة إطار العمل FrameworkClass Library (FCL). مكتبة الفئة القاعدية أصغر ، وتحتوي الميزات الأساسية الأعظم التي لا يستطيع البرنامج العمل بدونها . وهي تتضمن فقط تلك الفئات التي هي أساس واجب لدعم التطبيقات على إطار العمل إذا دعمت الميكروسوفت ذلك، تقول، نفذ إطار العمل إلى نظام تشغيل الكمبيوترات الشخصية Linux . إن مكتبة فئة إطار العمل أكبر، و تحتوي كل شيء آخر اعتقدت الميكروسوفت أنك تريد أن تملكه في برامجها ، ولكنه ليس أساسي بشكل جوهري ليدخل في مكتبة الفئة القاعدية . ويوجد فئات كثيرة جداً في مكتبة الفئة القاعدية ، بالآلاف من الفئات (نعم ، بالآلاف !) ، العدادات ، الواجهات ، وأنواع أخرى محتواة في مكتبة الفئة القاعدية ومكتبة فئة إطار العمل ، ستعتقد أنه من الصعب أن تجد فقط الفئة التي تحتاجها . لكن ليس ذلك صعباً ، على الأقل ليس صعباً جداً . يتضمن إطار عمل الدوت نت ميزة ما مسماة فضاءات الأسماء namespaces . تظهر جميع الأنواع في الدوت نت كهرم ـ تركيب مثل الشجرة ـ مع القليل فقط من المدخلات الدنيا عند الجذر . كل عقدة node في الهرم هي فضاء اسم . وتعرف أي فئة أو نوع آخر في المكتبات بتسمية جميع فضاءات الاسم بشكل وحيد ، من أسفل الجذر إلى فضاء الاسم المحلي الذي يحوي الفئة ، بفصل كل عقدة بنقطة (.) . بخلاف معظم الأهرام التي لديها جميع الفروع المبتدئة من عقدة جذر وحيدة ، لدى هرم فضاء اسم الدوت نت عقد جذور متعددة . يسمى فضاء اسم الجذر الأكبر النظام System . ويتضمن العديد من الفئات ، ولكنها تتضمن أيضاً عقد هرمية مباشرة متعددة (فضاءات اسم) . بما أن إطار العمل يتضمن ميزات لكل من تطوير تطبيق

قاعدة الويندوزWindows based و قاعدة الانترنت web-based ،فإنه يوجد فضاءات اسم تحتوي ميزات متطورة لويندوز محدد-Windows specificو لانترنت محدد web-specific . تظهر فضاءات الاسم فقط ضمن فضاء اسم النظام System، وتدعى Windows و Web . وجميع الكود المرتبط بالنماذج المعروضة على الشاشة في فضاء الاسم Windows يظهر في فضاء الاسم Forms ، وضمن فضاء الاسم هذا الفئة الفعلية التي تنفذ نموذج ما، مسمى Form . في الفيجوال بيسك ، تعرف فئة ما بتحديدها بكل فضاءات الاسم التابعة لها ، مبتدئاً من فضاء اسم الجذر التابع لها . فئة النموذج Form لديها الاسم المحدد الكامل التالي : System.Windows.Forms.Form

توجد جميع الفئات و الأنواع في مكان ما في الهرم ، رغماً من أنه لا تنحدر كل فئة من النظام.System . تظهر العديد من الميزات الداعمة المحددة للفيجوال بيسك في فضاء الاسم Microsoft.VisualBasic، الذي لديه "Microsoft " كعقدة جذر خاصة به بدلاً من "النظام System " عندما تنشئ مشاريع جديدة في الفيجوال بيسك ، الاسم للمشروع هو ،بقيمة ابتدائية، عقدة المستوى الأعلى top-level في الهرم . إذا أنشأت تطبيق Windows جديد مسمى WindowsApplication1 ، فإن نموذج القيمة الابتدائية "Form1" لديه الاسم المحدد الكامل التالي :

WindowsApplication1.Form1

اسم الفضاء للتطبيق الجديد ليس مجرد فئة ثانية second-class ملحقة منحدرة من فضاء الاسم System. يتم دمجه بشكل كامل في هرم فضاء اسم الدوت نت الكامل: فضاء الاسم WindowsApplication1 هو عقدة جذر ، فقط مثل عقد الجذر System و Microsoft ، يتضمن الفيجوال بيسك الميزات التي تدعك تغير القيمة الابتدائية لفضاء اسم تطبيقك ، أو مكان واحد من فئات التطبيق في فضاء اسم ما خاص . يمكنك حتى أن تضع فئات تطبيقك في فرع فضاء اسم النظام System . تغيير WindowsApplication1 إلى النظام.System.MySuperApp يتحرك من Form1 إلى:

System.MySuperApp.Form1

إذا كان تطبيقك بشكل فعلي عنصر أو مكتبة موجهة للاستخدام في البرامج ،فإن فئات تطبيقك ستظهر في فضاء الاسم الذي حددته عندما حمل البرنامج الآخر عنصرك في منطقة التطبيق الخاصة به.إن كودك سيبدو كجزء من فضاءات الاسم Microsoft-supplied . على الرغم من أنك تستطيع أن تضيف فئاتك إلى فضاء اسم النظام System،فإنك سوف تتعرض لغضب مبرمجي الدوت نت الآخرين . يتم فرض فضاء الاسم System ليكون ل"system" (اقرأ مزود الميكروسوفت) ميزات ، وهو ذلك . توجد أيضاً فرصة ما وهي أنه ربما يستخدم بائعان نفس مسار فضاء الاسم . لذلك ، لتجنب تناقضات فضاء الاسم المحتملة و النظرات الدنيئة من المبرمجين الآخرين ، عليك أن تسمي فئات تطبيقك مثل :

CompanyName.ApplicationName.ClassName

كفئة وحيدة أو نوع آخر لا يمكن أن يتم إفساده من خلال فضاءات الاسم المتعددة ، حتى ضمن نفس فرع الهرم . على أية حال ، يمكن أن يتقاسم نوعان أو فئتان الاسم المشترك في فضاءات اسم مختلفة ، حتى ضمن نفس الفرع .

تظهر جميع فئات مكتبة الفئة القاعدية ومكتبة فئة إطار العمل متمازجة في كل مكان في هرم فضاء الاسم الكامل . هذا يعني أنه لا يمكنك بالضرورة أن تقول فيما إذا كانت فئة مخصصة هي من مكتبة الفئة القاعدية أومن مكتبة فئة إطار العمل . بصراحة ، لا توجد مشكلة : فلن يهتم كودك من أي مكتبة تأتي فئة ما، طالما أنها متاحة للاستخدام على شبكة أجهزة المستخدم .

فقط قبل إصدار الفيجوال استديو 2008 ، أعلنت ميكروسوفت أنها ستجعل الكود المصدري أكثر توافقاً مع مكتبة فئة إطار العمل النسخة 3.5 المتاحة لمراجعة المطورين . وهذا يعني أن المبرمجين الذين يريدون معرفة كيف تصنف الميكروسوفت قائمة من أسماء ما في الذاكرة أو ترسم دائرة ملونة على نموذج ما والذي سيحصل على الأقل على نظرة جزئية عن كيفية عمل ذلك .

الكشوف والمجمعات Assemblies and Manifests

التجميع هو "وحدة نشر" لأجزاء تطبيق أو مكتبة الدوت نت . في 99.9 % من الحالات ، التجميع هو ببساطة ملف دوت نت قابل للتنفيذ (an .exe file) أو فئات مكتبة الدوت نت أو أنواع أخرى (a .dll file) . من الممكن أن تقسم التجميع بين ملفات متعددة ، لكن عادة هو ملف واحد من أجل مجمع واحد .إن ما يجعل ملف قابل للتنفيذ أو ملفات أخرى تجميع ما هو وجود كشف manifest . من أجل تجميعات ملف وحيد single-file، فإن الكشف manifest يظهر مباشرة في الملف : ويمكنه أن يظهر أيضاً في ملف ما بحد ذاته . الكشف هو قطعة ما كبيرة من البيانات التي تسجل تفاصيل هامة حول التجميع ، المتضمن اسمها ، معلومات إصدارها ، قيم ثقافتها المبدئية ، ومعلومات عن أنواعها و مجمعاتها الخارجية المرجعية ، و قائمة ما من جميع الملفات الموجودة في التجميع . لن تميز لغة التنفيذ المشتركة CLR تجميع ما بدون كشفه ، لذلك لا تفقده . يمكن أن تتضمن المجمعات اسم قوي strong name اختياري . وهذا يساعد على ضمان سلامة و أصل تجميع ما خلال توقيع رقمي متعلق بالكشف. يستخدم الاسم القوي مفتاح عام للكتابة السرية ليضمن أن التجميع منقطع النظير و لا يمكن التلاعب به. تتضمن الفيجوال استديو و إطار عمل الدوت نت أدوات تتيح لك إضافة اسم ما قوي للتجميع .

عندما تنشر تطبيقك ، سوف تعرض بشكل عادي جميع ملفات التجميع ، والملفات التشكيلية ، و أي ملفات متعلقة خاصة بتطبيقك في فهرس تنصيب التطبيق ، تماماً كما في الأيام الجوراسية القديمة قبل الدوت نت. صممت المجمعات المشاركة لكي يتم استخدامها من قبل أكثر من تطبيق واحد على آلة مفردة يمكن أن يتم تخزينها في ذاكرة التجميع العالمي المخفية Global Assembly Cache (GAC). يجب أن تملك جميع المجمعات المتركزة في ال GAC أسماء قوية . يمكن أن تسمح بعض الأنظمة فقط لمدير النظام أن يضيف التجميعات إلى ال GAC .

توصيف البيانات والمواصفات Metadata and Attributes

يتم إحضار المجمعات لك بالحرف *m* . بالإضافة إلى الكشوف و أعضاء نوع ما، تحتوي المجمعات أيضاً توصيف البيانات metadata. إن كود التطبيق و مكونات البيانات مخزنة في تجميع ما مطابق لبنود الكود والبيانات الموجودة في الفيجوال بيسك المتعلقة بالكود المصدري : ومن أجل كل نوع و عضو في كودك المصدري ، يوجد كود مساعد منفذ في التجميع المنشور . وهذا يحدث الإدراك ، و لا يوجد كثير من التغيير قبل انتشار الدوت نت المتطورة pre-.NET. ما الاختلاف الذي يجعل مترجم الفيجوال بيسك الآن يربط المعلومات الإضافية ـ توصيف البيانات ـ بكل نوع وعضو في التجميع. يُوَثِّق توصيف البيانات الاسم من المحتوى المساعد ، والمعلومات حول أنواع البيانات المطلوبة ، ومعلومات عن وراثة فئة لعنصر ، و رخص التأمين مطلوبة قبل أن يُستخدم العنصر من قبل المستخدم أو البرمجيات الأخرى . يعزز كودك المصدري للفيجوال بيسك توصيف البيانات من أجل أي عنصر لتجميعك من خلال المواصفات attributes . ينتج توصيف البيانات بوساطة مواصفة ما هي أكثر من مجرد عدد ID ما. تنفذ المواصفات فئات الدوت نت كاملة، مع المنطق المساعد و قيم البيانات الخاصة بها. يمكن لأي كود دوت نت والذي يعرف كيف يعالج المواصفات أن يفحص المواصفات من أجل عضو أو نوع ما و يتخذ حدث كما تدعي حاجته. وهذا يتضمن الفيجوال استديو ، ومترجم الفيجوال بيسك ، وتطبيقاتك التقليدية الخاصة بك . كيف يكون مثال ما عادي : يتضمن إطار عمل الدوت نت مواصفة ما مسماة ObsoleteAttribute تتيح لك هذه المواصفة أن تحدد أعضاء أو أنواع تجميعك وكأنها

مهجورة أو غير مدعومة منذ زمن طويل . (تستخدم الفيجوال استديو هذه المواصفة لتعرض تحذير ما كلما حاولتَ أن تستخدم ميزة ما خارجة عن التاريخ out-of-date لكل من مكتبة الفئة القاعدية أو مكتبة فئة إطار العمل). ولتستخدم المواصفة ، أضفها إلى عضو ما من تطبيقك مستخدماً أقواس زاوية الشكل (على الشكل <>):

```
Class MyClassWithOldMembers
        <ObsoleteAttribute()> Sub DoSomeWork()
        End Sub
    End Class
```

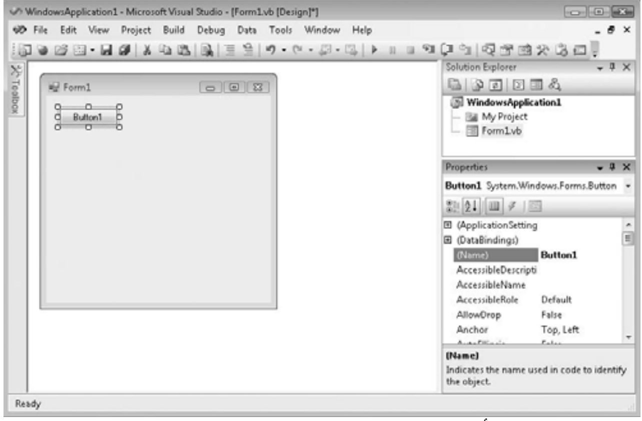
يعرف هذا الكود فئة ما وحيدة (MyClassWithOldMembers) بإجراء ذو عضو ما وحيد (DoSomeWork)، إجراء ما والذي يعمل بوضوح عمل ما. يتم ربط الإجراء بمواصفة ObsoleteAttribute. بواسطة الزبون ، تنتهي جميع أسماء المواصفة بالكلمة Attribute . *يمكنك* أن تهمل هذا الجزء من الكلمة إذا أردت ، طالما أن الكلمة الناتجة لا تتناقض مع أي كلمة محجوزة في لغة الفيجوال بيسك :

```
Class MyClassWithOldMembers
        <Obsolete()> Sub DoSomeWork()
        End Sub
    End Class
```

عندما تترجم الفئة و تخزنها في تجميع ما ،فإنه يتم تخزين المواصفة <ObsoleteAttribute> كجزء من تعريف DoSomeWork . يمكنك الآن أن تكتب تطبيق فيجوال بيسك منفصل والذي يمسح تجميع ما و مخرجات الاسم و حالات لكل نوع وعضو يجده ، وعندما يصادف ذلك البرنامج التحليلي العضو القديم ، سيكتشف ObsoleteAttribute في توصيف البيانات ، ويطبع الحالة :

```
DoSomeWork Procedure: Obsolete, don't use it!
```

يتم تصميم معظم المواصفات مع غرض ما خاص في العقل. تأمر بعض المواصفات الفيجوال استديو أن يعرض الأعضاء لفئة ما بطرق محددة . من المحتمل أنك عملت سابقاً مع ميزات تحرير النموذج form-editing للفيجوال استديو لتصمم تطبيق بسيط لسطح مكتب ويندوز عندما تضيف تحكم ما (مثل زر ما أو صندوق قائمة) لنموذج ما و تختار ذلك التحكم ، فإن الفيجوال بيسك ستتركك تحرر التفاصيل من ذلك التحكم من خلال منطقة لوحة الخاصيات Properties(انظر الشكل التالي).



يتم تنفيذ زر التحكم Button كفئة ما ، ويظهر العديد من أعضاء فئته في لوحة الخاصيات Properties ، ولكن ليس جميعها . عندما تم تصميم فئة الزر Button ، فإنه تتم إضافة المواصفات إلى أعضائه التي تخبر الفيجوال استديو أي الأعضاء يجب أن يظهر في لوحة الخاصيات Properties، وأي منها يجب أن لا يظهر . من الواجب على الفيجوال استديو أن تفحص هذه المواصفات ، و تعرض الخاصيات المطلوبة فقط.

التوافقية Versioning

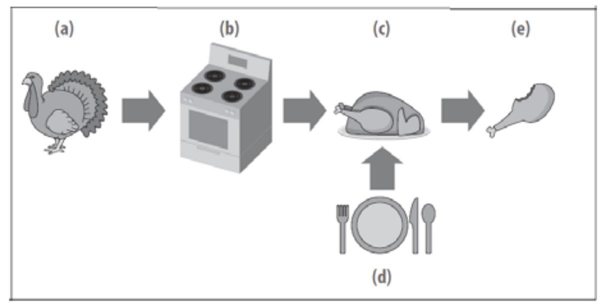
كما لديك ، تطبيقاتي سليمة من تحريراتها الأولية ، ولا أجد سبب لأعدلها أو أضيف ميزات لها . ولكن يوجد تنظيمات تطوير برمجة متضمنة شركة ضخمة واحدة لعمل ذلك ، ولكن لا أسبب إحراج ، سوف أشير إليها فقط بواسطة حرفها الأولي *M*— الذي يلتمس الحاجة إلى تفوق "one-up" المتنافسين بواسطة الخروج مع التوافقية المحسنة "improved" عن سابقاتها متخلية عن عروض البرامج . دعنا نقول أنه حدث الحرف "M" ليملك معالج كلمة شائعة ما والتي تتضمن توافقية 1.0 من فحص تهجئة عنصر ما. يحدث أيضاً الحرف "M" ليبيع أداة بريد ما و التي تعتمد بشكل محدد على التوافق 1.0 من نفس ذلك العنصر المشارك . إذا حرر الحرف "M"، في عرض رجل منافس ما ، تحديث ما لمعالج الكلمة و فحص تهجئة العنصر (الآن توافق 2.0)، فماذا يحدث لقدرة فحص تهجئة أداة البريد ؟

رغماً من أن هذا لا يحدث دائماً في الحياة الواقعية . ولكن إذا حدث ، التبديل لأساس عنصر ما متشارك مع عنصر أحدث. لكن يمكن أن يسبب توافق متضارب إلى حد ما مشاكل حقيقية . المشكلة المسرودة هي الانتشار لتوافقيات متعددة لعنصر ما منفرد على نفس شبكة الأجهزة ، جميعها في اتجاهات مختلفة . هل يمكن أن تحذف أي منها بشكل آمن ؟

تحل الدوت نت هذه المشاكل من خلال التوافقية versioning . *تعرف* جميع المجمعات التي تستخدم العناصر المشاركة بالضبط أي توافقية لعناصر مشاركة تطلبها . على الرغم من أنه يمكن أن يتم تصوير تطبيق ما ليستخدم توافقية فيما بعد ، فإنه سيستخدم التوافقية المحددة بشكل أصلي فقط من عنصر مشارك ما افتراضي . يمكن أن تتم إضافة التوافقيات المتعددة من عنصر منفرد مشارك ما إلى ذاكرة التجميع العالمي المخفية GAC، وتدعى ميزة ما جنباً إلى جنب side-by-sideبالانتشار deployment.تؤكد لغة التنفيذ المشتركة أن التطليق الصحيح يرتبط مع العنصر الصحيح .حتى أنه يمكنك أن تنفذ بنفس الوقت التطبيقات التي تستخدم توافقيات مختلفة لنفس العنصر.

من كود مصدري إلى ملف تنفيذي From Source Code to EXE

تعرف الآن بشكل جيد أن كل شيء موجود لتعرف أكثر حول الدوت نت لولا موضوع البرمجة المزعج. قبل التفتيش في كود فعلي ما ، دعنا نأخذ استراحة صغيرة ونفحص مدى عمر تطبيق ما ، من البداية إلى النهاية (انظر الشكل التالي):



1- إنك ، كمبرمج ، مسؤول عن تحضير العناصر الأساسية (a) من التطبيق . من أجل برامج الفيجوال بيسك ، هذا يعني إنشاء واحد أو أكثر من ملفات الكود المصدري مع امتداد (vb.) . يمكن أن تتضمن عناصرك أيضاً ملفات دعم أخرى ، مثل ملفات مصدرية (ملفات نصوص و رسومات ، وغالباً ما تستخدم دعم اللغات المتعددة).

2- يتم التلاعب بتطبيقك بوساطة مترجم الفيجوال البيسك كما في (b) . النتيجة هي تجميع ما ، يكمل مع كشف manifest ما و توصيف البيانات metadata. إن النتائج بالفعل هي لغة ميكروسوفت وسيطة MSIL شبه مترجمة وتتضمن الإصدار الجاهز للتنفيذ ready-to-execute من أنواع وأعضاء الكود المصدري الأصلي ، المتضمن كافة أسماء النوع و العضو . يمكن أن تكون كافة هذه المحتويات ناقصة الترجمة "decompiled" (معادة إلى اللغة الوسيطة الكاملة ، ورغماً من عدم إشباع الفيجوال بيسك) مستخدماً أداة ما مسماة ildasm.exe (مفكك Disassembler لغة الميكروسوفت الوسيطة) ، التي يتم إدخالها مع إطار عمل الدوت نت . بما أنه من المحتمل أنك لا تريد لأحد ما من أن يفكك تطبيقك وينظر إلى الكود ، فإن الميكروسوفت (أو أي طرف آخر) تدعم أيضاً obfuscator ، *الذي* يمزج بدرجة كافية المحتوى من كودك ليجعله صعباً بما فيه الكفاية حتى يستطيع إعاقة نظرات المتطفلين .

3- يتم نشر المجمع (c) إلى شبكة أجهزة المستخدم . يتم استخدام طرق مختلفة قليلة لتنشر التطبيق ، المتضمن (1) إنتاج حزمة تنصيب مثبت الويندوز القياسي ، (2) إنتاج توزيع ما ClickOnce : أو (3) إنجاز تنصيب نسخ ما xcopy ، والذي لا يتضمن شيء أكثر من نسخ المجمع التنفيذي نفسه إلى الجهاز الهدف . لايوجد مشكلة في أي طريقة نشر تختارها ، وإن وقت التنفيذ NET runtime. الخاص بالدوت نت في (d) يجب أن يتم تثبيته على شبكة أجهزة المستخدم .

4- المستخدم يأكل -أقصد أنه يشغل - البرنامج كما في (e) . تعمل لغة التنفيذ المشتركة لمجمع لغة ميكروسوفت الوسيطة ترجمة نهائية على الفور-just-in time (JIT)، لتحضرها من أجل الاستخدام على المنصة المحلية . ومن ثم تعرض التطبيق للمستخدم ، وتدير جميع سمات التطبيق عندما يكون مشغل . يختبر المستخدمون مستوى المتعة والرضا المصادف بشكل قليل عند استخدام تطبيقات برمجية أخرى .

تقديم الفيجوال بيسك Introducing Visual Basic

قواعد المنطق و البيانات The Basics of Logic and Data

كمثال ما، دعنا نقول أنك تلقيت طلب ما من قسم بيع البرامج لعمل برنامج يقلب جميع الحروف في أي قطعة كبيرة من نص مجهز للبرنامج .لذلك أولاً اكتشف المنطق ، وثم نفذه في الفيجوال بيسك . باستخدام شبه الكود(الكود المزيف) pseudocode لغة برمجة اصطناعية (مفتعلة)التي تركبها بنفسك لتساعدك على كتابة البرامج)، بإمكانك أن تضع مخطط (أو مسودة أولية sketch) خارج هذه المهمة الأساسية (مع ترقيم الأسطر الرئيسية):

1- احصل على النص الأصلي (أو سلسلة حرفية) من المستخدم .
2- إذا لم يزودك المستخدم بأي محتوى ، عندئذ اخرج حالاً .
3- حضر مكان ما للسلسلة الحرفية المعكوسة ، فارغ حالياً .
4- كرر التالي حتى تكون السلسلة الحرفية فارغة :
5- انسخ الحرف الأخير من السلسلة الحرفية الأصلية المتبقية ،
6- ضع ذلك الحرف على نهاية السلسلة المقصودة .
7- قصّر السلسلة الأصلية ، بإسقاط الحرف الأخير .
8- [أنهي قسم التكرار]
9- دع المستخدم يرى السلسلة الحرفية المقصودة .

يمكنك أن تكتب هذا المنطق بطرق متعددة : وهذا مجرد مثال واحد فقط . يمكنك الآن أن تحول الشبه شفرة هذه إلى لغة من اختيارك: في هذه الحالة ، الفيجوال بيسك (لا تقلق حول تفاصيل القواعد اللغوية حتى الآن):

```
1:      originalText = InputBox("Enter text to reverse.")
2:      If (Len(originalText) = 0) Then Return
3:      finalText = ""
4:      Do While (originalText <> "")
5:          oneCharacter = Right(originalText, 1)
6:          finalText &= oneCharacter
7:          originalText = Left(originalText, _
         Len(originalText) - 1)
8:      Loop
9:      MsgBox("The reverse is: " & finalText)
```

إن هذا الكود المصدري *source code* جاهز الآن ليتم استخدامه في برنامج الفيجوال بيسك . و يوضح أيضاً السمات الأساسية المتعددة لكتابة الكود :

- تدعى الخطوات المفردة من التعليمات التدريجية step-by-step عبارات statements . و في الفيجوال بيسك،تظهر كل عبارة على سطر ما بنفسها . يمكنك قطع العبارات الطويلة إلى سطور متعددة بوصل السطور بفراغ مزدوج تحت كلمة ما space-underscore، كما هو واضح في السطر 7 من الكود .عندما يتم نشر عبارة منفردة عبر سطور متعددة في هذه الحالة ، فإن العبارة الكاملة تدعى أحياناً بسطر منطقي *logical line*.بما أنه غالباً ما يتضمن سطر منطقي مفرد ما عمل فيجوال بيسك ابتدائي مفرد ما فقط (مثل عمل If أو Do ، أو استخدام أعمال الإسناد المختلفة لإشارة المساواة [=]) ، ويشار أيضاً إلى هذه الأعمال كعبارات.
- تتم معالجة عبارات الكود كل واحدة بدورها، من الأعلى إلى الأسفل . على أي حال ، تبدل عبارات محددة التدفق العادي للبرنامج من الأعلى إلى الأسفل top-to-bottom ، كما تم عمله بالكتلة Do While...Loop على السطر 4 و السطر 8 من الكود النموذج. وهكذا تدعى العبارات بعبارات ضبط التدفق flow control ، و تتضمن الحلقات (تكرار قطعة الكود)، والشروط (بشكل اختياري معالجة قطعة ما من الكود المؤسس على مقارنة ما أو نتيجة محسوبة) ، والقفزات (تحريك بشكل مباشر إلى قسم آخر من الكود) .
- يتم تخزين البيانات في المتغيرات *variables* ، والتي تسمى حاويات لقيم البيانات . تتضمن قطعة الكود النموذج ثلاث متحولات وهي : originalText، oneCharacter، finalText ، تخزن جميع هذه المتغيرات نص (سلسلة حرفية) بيانات . يسمح لك نظام النوع المشترك للدوت نت (CTS) بأن تنشئ متحولات لأربع أنواع رئيسية من قيم البيانات الأساسية وهي: نصية text(كل من الحروف المفردة و السلاسل الحرفية الطويلة) ، أعداد numbers(كل من القيم الصحيحة والعشرية) ، التواريخ dates(و الأوقات)، المنطقية Booleans(قيم صح أو خطأ) . يمكنك أيضاً أن تنشئ أنواع معقدة أكثر من البيانات بتجميع الأنواع الأساسية .
- تُخزن البيانات في متحول ما من خلال إسناد ما.assignment . يتضمن هذا بشكل عام توظيف اسم متغير ما على جهة اليسار من عملية إسناد ما assignment operator (=) ، و وضع البيانات أو الحسابات لتخزن في ذلك المتغير على جهة اليمين من نفس إشارة المساواة تلك . تخزن العبارة " " = finalText على السطر 3 سلسلة حرفية ما فارغة ("") في المتغير finalText.وتعرض عبارة الإسناد =& على السطر 6 قواعد إسناد لغوية مختلفة نوعاً ما .
- يمكن أن تتضمن العبارات استدعاءات الدالة function calls ، وقطع من التخصص الوظيفي المسبق الكتابة ، وتقع جميعها تحت اسم مفرد . تعمل استدعاءات الدالي حزمة عمل ، و بعدئذ تعيد return نتيجة نهائية ، وقيم بيانات . يتم إتباع أسماء الدالي بمجموعة من الأقواس ، التي يمكن أن تتضمن الصفر أو معاملات نسبية arguments أكثر ، وقيم بيانات شرطية مدعومة بكود الاستدعاء الذي يستخدمه الدالي ليولد نتائجه . يتضمن الكود النموذج أمثلة عديدة من استدعاءات الدالي ، والمتضمنة الدالي Right على السطر 5 . يعيد هذا الدالي نسخة ما من الحروف التي على أقصى اليمين rightmost من سلسلة نصية أخرى . إنه يوافق وسيطين اثنين : السلسلة الأصلية من التي استخرجت الحروف من أقصى اليمين ، و قيمة صحيحة ما تشير إلى عدد الحروف اللازمة لتعود . يعيد الكود Right(originalText, 1) نسخة ما من الحرف المفرد على أقصى اليمين (1) من originalText.

عندما تستخدم دالي ما في كودك المصدري ، عندئذ يمثل الدالي شيء صغير كمتغير: جميع النصوص لاستدعاء الدالي ، من البداية لأسمائها إلى النهاية بأقواسها المغلقة ، يمكن تُستبدل بمتغير ما والحاوي على نفس البيانات الناتجة. لا تظهر استدعاءات الدالي على جانب الجهة اليسرى من عبارة إسناد ما ، ولكن يمكن أن تظهر تقريباً في أي مكان آخر والذي يمكن أن يظهر فيه المتغير . على سبيل المثال ، يمكن أن يستخدم السطران التاليان لاستبدال السطر 2 في النموذج :

```
' Replacing --> If (Len(originalText) = 0) Then Return
lengthOfText = Len(originalText)
If (lengthOfText = 0) Then Return
```

- بالإضافة إلى الدوال ، تتضمن أيضاً الفيجوال بيسك إجرائيات *Procedures*. تحزم الإجرائيات *Procedures* الكود المكتوب مسبقاً في حزمة معرفة ، تماماً كما في الدوال functions. ولكنها لا تعيد قيمة. يجب أن يتم استخدامها كعبارات مستقلة : ولا يمكنك أن تستخدمها حيث ستستخدم استدعاء متغير ما أو دالي ما . الاستدعاء إلى MsgBoxعلى السطر 9 هو مثال نموذجي لاستدعاء الإجرائية في الاستخدام . (MsgBoxهو دالي فعلي .

يمكن أن يتم جعل الكود النموذجي المدرج سابقاً أكثر فعالية قليلاً . في الحقيقة ، من المحتمل بشكل كامل أن الميكروسوفت حصلت على مسودة أولية من هذا الكتاب ، بما أنها تضمنت ميزة سلسلة الحروف المقلوبة string-reversal اليمينية في الفيجوال بيسك ، و سمتها StrReverse :

```
originalText = InputBox("Enter text to reverse.")
If (Len(originalText) = 0) Then Return
finalText = StrReverse(originalText)
MsgBox("The reverse is: " & finalText)
```

ذلك صحيح : تضمنت الفيجوال بيسك سابقاً ميزة السلسلة الحرفية المقلوبة ، والبعض من كود المكتبة المكتوب مسبقاً الذي أناقشه .تتضمن الفيجوال بيسك الكثير مثل الدوال الجوهرية intrinsic functions التي تعتبر جزء من اللغة ، و التي تحزم التخصص الوظيفي المفيد المكتوب مسبقاً. يظهر العديد من هذه الدوال في فضاء الاسم.Microsoft.VisualBasic ، والذي يتم صنعه بشكل أوتوماتيكي متيسر لكودك المصدري الخاص بالفيجوال بيسك عندما تنشئ مشروع فيجوال بيسك جديد .

المتغيرات و أنواع البيانات Data Types and Variables

عملياً ما أطلبه من تطبيق الفيجوال بيسك الخاص بي ليعمل هو التالي : خذ البيانات من مصدر ما (لوحة المفاتيح ، القرص الصلب ، الانترنت ، إلخ) ومثلها بطريقة مفيدة ما. ستدير جميع البرامج التي كتبتها بشكل فعلي على الأقل بعض البيانات في الذاكرة . تخزن كل قيمة للبيانات في منطقة محددة ما من ذاكرة الكومبيوتر ، على الرغم من أنها محددة بواسطة لغة التنفيذ المشتركة Common Language Runtime. توجد العبارات في الفيجوال بيسك بشكل رئيسي لتدير و تعالج هذه البيانات بطرق معقدة و مفيدة .

يتم تخزين كل البيانات المدارة بواسطة لغة التنفيذ المشتركة في ذاكرة الكمبيوتر ، مع كل قيمة للبيانات منفصلة و محمية عن القيم الأخرى . وكأن لدى كل قيمة بيانات كوب شاي مستقل خاص بها كما يبدو في الشكل:



تمتلك جميع قيم البيانات المدارة بواسطة لغة التنفيذ المشتركة محتوى *content* ونوع *type*. المحتوى هو البيانات الفعلية : السلسلة النصية "abc,"، العدد 5 ، فاتورة بيع ، شاي بيكو البرتقالي اللون . كل ما تضعه في كوب الشاي ، وذلك هو المحتوى. في بعض الحالات ، تسمح لك الدوت نت أن تخزن بدون شك لاشيء nullable في كوب الشاي (من أجل الأنواع المرجعية كما موصوفة باختصار، أو أنواع قيمة ملغاة أو باطلة) .

يشير النوع إلى النوع المحتوى المخزن في كوب الشاي . في الشكل السابق ، هذا معروض بواسطة الشكل لكل كوب شاي . لكل كوب شاي حدود على النوع من البيانات التي يمكن أن يتم سكبها في كوب الشاي : سلسلة نصية ، عدد صحيح ما ، فاتورة زبون ما .

المحارف Literals

بعض قيم البيانات الأساسية ،مثل الأعداد و السلاسل النصية، يمكن إدخالها في كودك المصدري و استعمالها كما هي . على سبيل المثال ، يعرض الإجراء MsgBox نافذة برسالة نصية مدعومة . تتضمن العبارة : MsgBox("The answer is " &" 42) سلسلة محرفية *literal string*، " The answer is " و قيمة صحيحة محرفية ، 42 . (الرمز "&" هو معامل يصل قيمتين مع بعضهما البعض في سلسلة حرفية جديدة .) تستخدم المحارف مرة ، ومن ثم تهلك . إذا أردت أن أعرض نفس الرسالة مرة أخرى ("The answer is 42") ، علي أن أكتبها مرة ثانية بنفس قيم المحارف في جزء مختلف من الكود المصدري .

تدعم الفيجوال بيسك أنواع عديدة من المحارف الأساسية. تحاط المحارف النصية *String literals* دائماً بعلامات اقتباس quote marks. إذا أردت أن تدخل علامة اقتباس ما لوحدها في المنتصف لنص ما، يتضمن علامتين بدلاً من علامة واحدة : "This is ""literally"" an example."

يمكن أن تكون المحارف النصية String literals حقاً ، حقاً طويلة ، تصل حتى حوالي 2 بليون حرف بالطول : إذا أردت أن تكتب حرف واحد فقط بكل ثانية ، فإنه سوف يستغرق أكثر من 63 سنة ليصل إلى طول السلسلة الأعظمي . تتضمن الفيجوال بيسك أيضاً المحارف الحرفية character literal التي هي بالضبط حرف واحد في الطول : إذا أردت أن تكتب حرف واحد فقط بكل ثانية ، حسناً ، لايهم . يتم تمييز هذه المحارف الحرفية بواسطة التذييل "c"بعد النص . يتم إدخال محرف الحرف "A" كما : "A"c
يتم إحاطة محارف الوقت والتاريخ بإشارات عدد number signs بدلاً من علامات الاقتباس . الوقت أو التاريخ (أو كلاهما) الذي تدخل يمكن أن يتم تمييزه في أي تنسيق من قبل الميكروسوفت ويندوز في منطقتك المحددة . إذا كنت مستخدماً الفيجوال استديو Visual Studio ، فإنه سوف يعيد تنسيق تاريخك عندما تكتبه في المحرف :#7/4/1776#
يكوّن أحد عشر نوعاً مختلفاً من قيم البيانات العددية ـ كل من القيم العددية الصحيحة و ذات النقطة العائمة floating-point ـ جوهر "core"المجموعة من أكواب الشاي العددية. و من يحتاج إلى أكثر من 11؟ مع الأحد عشر كوب شاي تلك، يمكنك أن تدير الأعداد من الصفر بجميع الطرق إلى 1×10^{300} و ما بعد ذلك أيضاً . ولتستخدم المحرف العددي، اكتب العدد على اليمين في كودك ، مثل 27 ، أو 3.1415926535 . تتيح لك الفيجوال بيسك أيضاً أن تحدد أي من الأحد عشر كوب شاي عددي تستخدم من أجل عدد ما ، بإلحاق حرف خاص إلى نهاية العدد . وعادة 27 هو عدد صحيح 27 . ولتجعله بدقة العملة "عشري "decimal,""، مذيل بإشارة at (@):27@ عندما أناقش أنواع البيانات بالتفصيل الكامل ، سوف أسجل في قائمة الحروف الخاصة المختلفة ، مثل (@) ، التي تدون نوع البيانات للأعداد الحرفية . النوع الرابع والأخير من محارف الفيجوال بيسك هو المحرف المنطقي Boolean .
تمثل القيم المنطقية النوع الأبسط من بيانات الكمبيوتر : البت . القيم المنطقية هي إما صح أو خطأ ، موصول أو مقطوع ، نعم أو لا ، لذيذ أو مقرف ، قطط أو كلاب ، صفر أو غير الصفر . يمثل المنطق Booleans دائماً أحد حالتين أو قيمتين متعاكستين . تمثل اللغة التي استخدمها العالم جورج بول العبارات المنطقية كمعادلات رياضية . الكمبيوترات تحب *love* الجبر المنطقي .لذلك جميع العمليات الأساسية للكمبيوتر ، مثل الجمع ، يتم تنفيذها باستخدام التخصص الوظيفي المنطقي .
تتضمن الفيجوال بيسك المحارف المنطقية صحTrue أو خطأ False. لا يوجد علامات اقتباس ، ولا إشارات أعداد ، يوجد فقط الكلمات صح *True* وخطأ *False*. في حالات محددة ، يمكنك معالجة الأعداد كقيم منطقية Boolean values. عليك أن تعرف الآن أن خطأ تعادل الصفر (0) ، و صح تعادل كل شيء آخر (مع أنه بشكل عام ،يستخدم -1 من أجل كل شيء آخر"everything else") .

المتغيرات Variables

جميع قيم البيانات المحرفية جيدة ، ولكنها مفيدة مرة واحدة فقط ، و بعدئذ ترحل .في كل مرة تريد أن تستخدم قيمة محرفية ، عليك أن تعيد كتابتها .وكأن قيم البيانات مخزنة في الأكواب المعدة للطرح بعد الاستخدام بدلاً من أكواب الشاي الصينية الجميلة . و بالإضافة إلى ذلك ، يدخل المبرمجون فقط القيم الحرفية ، وليس المستخدمين ، لذلك فهي محدودة الاستخدام بالنسبة لبيانات المستخدم المدارة.
ببساطة المتغيرات *Variables* ليست أكواب معدة للطرح بعد الاستخدام : يمكن إعادة استخدامها . يمكنك المحافظة على وضع نفس النوع من الشاي مرة بعد مرة في كوب الشاي . كوب الشاي متغير نصي يمكنه أن يواصل سلسلة ما من أجل إعادة الاستخدام مرة بعد مرة . على سبيل المثال ، في هذه القطعة من الكود ، يستجيب المتغير للسلاسل المختلفة المسندة إليه :

```
       Dim response As String
1:      response = "A"
2:      MsgBox("Give me an 'A'!")
3:      MsgBox(response)
4:      MsgBox("Give me another 'A'!")
5:      MsgBox(response)
6:      MsgBox("What's that spell?")
7:      response = StrDup(2, "A")
8:       MsgBox(response)
```

يتم إسناد المتغير response مرتين بنصين مختلفين : "A" (في السطر 1) وبعدئذ "AA" (في السطر 7) . إنه يحافظ على كل قيمة تم إسنادها إليه أخيراً : يعرض كل من السطرين 3 و 5 "A" في نافذة صندوق رسالة ما . و ليس عليك أن تسند فقط النصوص المحرفية إليه (المتغير) : يمكن لأي شيء يولد النص أن يسند نتيجته للمتغير response. يستخدم السطر 7 دالة فيجوال بيسك جاهزة ، StrDup ، ليعيد سلسلة الحرفين two-character "AA" و يسندها إلى المتغير response. إن استخدام المتغيرات هو عملية ذات خطوتين two-step . أولاً عليك أن تصرح *declare* عن المتغير ، و بعدئذ أن تسند *assign* قيمة ما إليه . تهتم العبارة Dim بقسم التصريحات declaration: وهو يدعك تشير إلى كل من الاسم *name* والنوع *type*للمتغير . وقواعده النحوية syntax صريحة إلى حد ما .

```
 Dim response As String
```

حيث responseهو الاسم للمتغير و String هو نوعه . يحدث التصريح باستخدام عملية المساواة = كمايلي :

```
response = "The answer"
```

يمكن أن يملك متغير مفرد قيم جديدة مسندة إليه بشكل متكرر . لهذه المرات عندما تريد لمتغيرك أن يملك قيمة محددة ما مباشرة حين التصريح ، فإنه بإمكانك أن تدمج التصريح و الإسناد داخل عبارة منفردة :

```
Dim response As String = "The answer"
```

بالطبع ، إنك غير مقيد بتصريح مفرد فقط : يمكنك أين تنشئ متحولات عديدة كما تحتاج في كودك . ويستخدم عادة كل واحد عبارة Dim الخاصة به :

```
Dim question As String
Dim answer As String
```

بإمكانك أيضاً أن تجمع هذين المتحولين في عبارة وحيدة :

```
Dim question As String, answer As String
```

وهذه هي تماماً البداية لما هو ممكن مع العبارة Dim .

الأنواع المرجعية و الأنواع ذات القيمة Value Types and Reference Types

تخزن متغيرات النوع ذو القيمة قيمة فعلية ما: الشاي في كوب شاي نوع ذو قيمة هي المحتوى (المضمون) بذاته . جميع قيم البيانات المحرفية التي ذكرتها سابقاً ، ما عدا النصية Strings ، هي أنواع ذات قيمةvalue types . تخزن متغيرات النوع المرجعي *reference types"* "مرجع ما " أو إشارة إلى البيانات الفعلية ، توجد البيانات في مكان ما آخر من الذاكرة . عندما تمعن النظر في كوب شاي قيمة مرجعية ، عليك أن تعرف أوراق الشاي في الأسفل لتحدد المكان الذي تسكن فيه البيانات . إما تملك الأنواع المرجعية البيانات أو لا تملكها . في حال غياب البيانات ، يأخذ نوع مرجعي ما القيمة لاشيء Nothing ، وهي كلمة محجوزة ما للفيجوال بيسك والتي تشير إلى أنه لا بيانات . إن الأنواع ذات القيمة ليست لا شيء Nothing:أبداً فهي تتضمن دائماً بعض القيم ، من المحتمل أن القيمة الافتراضية defaultلذلك النوع (مثل الصفر من أجل الأنواع العددية). يتيح لك نوع "بدون قيمة nullable"خاص أن تسند لاشيء Nothing إلى نوع ما ذي قيمة ، متيحاً لك أن تنفذ نفس المنطق " هل توجد أي بيانات هنا بأي حال من الأحوال " الذي يوجد مع الأنواع المرجعية . سوف أناقش الأنواع بدون قيمة nullable لاحقاً.

أنواع البيانات Data Types

إن نوع البيانات النصية مفيد ، و لكنه مجرد واحد من أشكال أكواب الشاي المتوفرة لديك . يعرف إطار عمل الدوت نت أنواع البيانات data types جوهرية متعددة. يتم تنفيذ كل نوع بيانات كفئة محددة ضمن فضاء اسم النظام System . نوع البيانات الأساسية الأهم ، كوب الشاي الكبير الذي يستطيع أن يملك أي نوع من البيانات ، ويسمى بالكائن Object. أكثر من مجرد كائن ما، فهذا كائن بحرف *O* كبير.في هرم فضاء أسماء مكتبة فئة الدوت نت ، فإنه يُحدَد في System.Object . وهو الأم لجميع الفئات في الدوت نت : وجميع الفئات الأخرى ، والتراكيب ، والعدادات ، و التفويضات ، لا مشكلة في أي مكان يسكن كل منها بالنسبة لهرم فضاء الاسم ، فيما إذا تمت كتابته من قبل الميكروسوفت أو من قبلك ،فهي مشتقة في النهاية من System.Object. ليس هناك ما يستوعبه : لايمكنك أن تنشئ نوع ما يشتق بشكل أخير من أي شيء آخر . لذلك عد إلى أنواع البيانات تلك "الجوهرية"التي أشرت إليها سابقاً . والتي تربط الأنواع الأربعة من قيم البيانات المحرفية التي جمعتها من قبل وهي : النصوص، التواريخ ، الأعداد ، و القيم المنطقية . ويصنف الجدول التالي أنواع البيانات الجوهرية تلك . يملك كل نوع أيضاً اسم فيجوال بيسك محدد والذي يمكن (و يجب) أن يستخدم بدلاً منه .

الجدول 2-1 . أنواع بيانات جوهر الدوت نت و الفيجوال بيسك *Visual Basic data types*.

اسم VB	اسم NET	الوصفDescription
Boolean	Boolean	يدعم نوع البيانات المنطقية فقط قيم صح Trueو خطأFalse . من الممكن أن تحول الأعداد إلى قيم منطقية : الصفر 0 يقابل خطأ و كل شيء آخر يصبح صح .وعندما تحول منطق ما بإعادته إلى عدد ،فإن خطأ يصبح 0 و صح يصبح 1- .
Byte	Byte	نوع بيانات عددي ما ، يخزن البايت بايت وحيد (8- بت) أعداد صحيحة بدون إشارة ، تتراوح من الصفر إلى 255.إن نوع بيانات البايت مفيد جداً للعمل مع بيانات غير نصية، مثل الصور الرسومية .
Char	Char	يخزن نوع البيانات الحرفية Charبالضبط حرف نصي واحد . تمثل كل قيمة بيانات حرفية 2 بايت (16 بت) من التخزين ، لذلك يستطيع أن يدير مجموعة حروف ذات بايت مضاعف ، مزودة الدعم للغات كاللغة اليابانية التي تملك عدد كبير من الحروف . على الرغم من أنه اعتاد أن يخزن حروف نصية مفردة ، يحفظ نوع البيانات الحرفية الحروف كقيم عددية صحيحة ، متراوحة من 0 إلى 65,535 وذلك بشكل داخلي.
Date	DateTime	يعالج نوع بيانات الوقت والتاريخ جميع التواريخ بين كانون الثاني 1،1 بعد الميلاد و31 كانون الأول، 9999 بعد الميلاد،في التقويم الغريغوري.يتم إدخال الوقت بشكل جيد : إذا لم يتم تحديد الوقت ، يستخدم وقت منتصف الليل .يخزن نوع بيانات التاريخ داخلياً التاريخ والوقت كعدد من " العلامات الصغيرة" منذ منتصف الليل في 1،1 كانون الثاني بعد الميلاد . كل إشارة هي جزء من100 بليون من الثانية (نانو بالثانية).
Decimal	Decimal	يتم تصميم نوع البيانات Decimal بالاعتماد على العملة . إنها دقيقة جداً في الحسابات الرياضية ، ولها مجال جيد جداً ، يدعم الأعداد التي هي بعد 79- الذي يتبع ب 27 صفر ، الموجبة و السالبة ، (هل قلت 79- متبوع ب 27 صفر؟) وذلك طوله 29 رقم ، و ذلك هام لتتذكر،نظراً لذلك حصلت على 29 رقم إجمالاً على كلا جانبي من الفاصلة العشرية. يأتي العدد 79- المتبوع ب 27 صفر مع التقييد بأنه لا أرقام إلى اليمين من الفاصلة العشرية . إذا أردت موقع عشري واحد ، عليك أن تترك واحد إلى اليسار (الجزء العشري من اللوغاريتم) و أن تبقي الأعداد إلى 9.7 ـالمتبوع ب 27 صفر . إذا أردت 29 رقم بعد العدد العشري ، لقد حصلت على صفر ضخم كبير من أجل اللوغاريتم . إذا كنت قد اعتدت أن تستخدم الفيجوال بيسك 6.0 ،العدد العشري هو مشابه لنوع البيانات الثانويCurrency .
Double	Double	يعالج نوع البيانات المضاعف Double الأعداد الممكنة الكبرى من جوهر أنواع البيانات العددية . ومجاله هو حوالي 4.94×10^{-324} إلى $1.798 \times 10^{+308}$ للأعداد الموجبة ، وبمجال مشابه لقيم الأعداد السالبة . بينما أنت تفكر أنك في سماء الأعداد الضخمة giganto-number، نوع البيانات المزدوج (المضاعف) هو جهراً غير دقيق في الحسابات المعقدة . في بعض الأوقات الحساب الذي يجب أن يؤدي إلى الصفر سوف يحسب بدقة كشيء مثل 0.0000000000005434 ، الذي تم إغلاقه . لكن مقارنة هذه الأعداد مع الصفر سوف تفشل ، بما أنها ليست صفراً .
Integer	Int32	إن نوع البيانات الصحيحة Integerهو 4 بايت (32 بت) نوع صحيح ذو إشارة .

		وهو يعالج الأعداد من -2,147,483,648 إلى 2,147,483,647 . إذا كنت مبرمج فيجوال بيسك سابق في الدوت نت ،فإن هذا نوع بيانات صحيحة جديدة مكافئ لنوع البيانات الصحيحة Longذات الإصدار 6.0 .
Long	Int64	نوع البيانات الصحيح الطويل Long أكبر من الصحيح Integer: وهو 8 - بايت (64 - بت) نوع صحيح ذو إشارة. و يعالج الأعداد من –9,223,372,036,854,775,808 إلى 9,223,372,036,854,775,807 . وهو ليس نفس النوع الصحيح الطويل للفيجوال بيسك 6.0 القديم ، وكما لديه سعة تخزين مضاعفة .
Object	Object	الكائن هو نوع جوهري لجميع أنواع الدوت نت . وهو يأخذ مكانه عند قمة هرم النوع و الفئة: وهو الفئة الأساسية الجوهرية لجميع الفئات الأخرى. وهو نوع مرجعي ، على الرغم من أن الأنواع ذات القيمة بشكل أخير تشتق منه ، أيضاً .
SByte	SByte	نوع البيانات العددية ، يخزن SByte بايت وحيد single-byte(8 بايت) لأعداد صحيحة بدون إشارة ، المتراوحة من –128 إلى 127 . وهو إصدار ذو إشارة من نوع بيانات البايت Byte التي بدون إشارة .
Short	Int16	نوع البيانات القصيرة Shortهو 2- بايت (16 - بت) نوع أعداد صحيحة ذات إشارة . وهو يخزن الأعداد من –32,768 إلى 32,767 إذا كنت مبرمج فيجوال بيسك سابق في الدوت نت ، فإن نوع البيانات Shortالجديد هذا مكافئ لنوع البيانات الصحيح Integerللإصدار 6.0.
Single	Single	يشبه نوع البيانات المفرد Singleكثيراً نوع البيانات المضاعف Double، فقط أصغر مجاله للأعداد الموجبة حوالي 1.4 × 10 $^{-45}$ إلى 3.4 × 10 $^{+38}$ ، مع مجال مشابه من أجل الأعداد السالبة . مثل نوع البيانات المضاعف Double، يعاني نوع البيانات المفرد Singleنوعاً ما عدم الدقة خلال العمليات الحسابية .
String	String	نوع البيانات الحرفي هو نوع مرجعي و يصل تخزينه إلى حوالي 2 بليون حرف من النص. ويخزن حروف شبه الكود ، التي تكون قدرتها 2- بايت (16- بت) لتخزين الحروف من معظم اللغات في العالم ، المتضمنة لغات ذات أبجدية ضخمة ، مثل اللغة الصينية .
UInteger	UInt32	تخزن الأعداد الصحيحة UIntegerبدون إشارة 4- بايت (32- بت) أعداد صحيحة بدون إشارة ، متراوحة بين 0 إلى 4,294,967,295 . وهو إصدار بدون إشارة لنوع البيانات الصحيحة Integerذات الإشارة .
ULong	UInt64	يخزن النوع الصحيح الطويل بدون إشارة ULong8- بايت (64بايت) أعداد صحيحة بدون إشارة ، متراوحة من 0 إلى 18,446,744,073,709,551,615 . وهو إصدار بدون إشارة لنوع البيانات الطويلة Longذات الإشارة .
UShort	UInt16	يخزن النوع الصحيح القصير بدون إشارة UShort2- بايت (16بايت) عدد صحيح بدون إشارة ، متراوحة من 0 إلى 65,535 . وهو إصدار بدون إشارة لنوع الصحيحة القصيرة Shortذات الإشارة .

إن مطوري الميكروسوفت مسؤولون عن أنواع بيانات الفيجوال بيسك التي تنهي بشكل جيد تلك المهمة بما أن جميع أنواع بيانات الفيجوال بيسك الجوهرية هي ببساطة أغلفة من أجل أنواع بيانات محددة منفذة من قبل الدوت نت . أسماء الفيجوال بيسك المعطاة لكل من هذه أنواع البيانات الجوهرية هي بشكل كامل قابلة للتبادل مع أسماء الدوت نت . فعلى سبيل المثال ، النوع الصحيح Integerمكافئ بشكل كامل للنظام الصحيح 32 System.Int32. في الحقيقة ، عند كتابة كود الفيجوال بيسك ، من الأفضل أن تستخدم مرادفات الفيجوال بيسك ، بما أن معظم مطوري الفيجوال بيسك يتوقعون أسماء نوع البيانات هذه في الكود الذي قرؤوه وكتبوه .

ماعدا الكائن Objectو النص String ، جميع أنواع البيانات هذه هي أنواع ذات قيمة . تشتق جميع الأنواع ذات القيمة من System.ValueType (الذي بدوره يشتق من System.Object) . البايت ذو الإشارة SByte ، الصحيح بدون إشارة UInteger ، الصحيح القصير بدون إشارة UShort، الصحيح الطويل بدون إشارة ULong ، وأنواع بيانات تم إضافتها إلى الفيجوال بيسك مع الإصدار 2005 ، على الرغم من أن مكافئات فضاء أسماء النظام System الخاص بها هي في الدوت نت منذ بدايته. على خلاف أنواع البيانات الجوهرية الأخرى ، فإن هذه الأنواع الأربع ليست "خاضعة لتوصيف اللغة المشتركةCLS-compliant " . وهذا يعني أنه، لا يمكن استخدامها لتتفاعل مع عناصر الدوت نت و اللغات التي تعرف نفسها بأنها ميزات الدوت نت الجوهرية فقط المطلوبة كثيراً . على العموم هذا التقييد ليس بالكثير ، ولكن كن على حذر عند العمل مع مكونات أو لغات ثانوية.

التصريحات المتقدمة Advanced Declaration

عندما ذكرت الحاجة إلى تصريح *declaration*و إسناد المتحولات ، كنت أركز فعلاً على الأنواع ذات القيمة . تتطلب الأنواع المرجعية خطوة إضافية واحدة : وهي الاستنساخ *instantiation*.إليك عبارات التصريح التالية :

```
Dim defaultValue As Integer
        Dim nonDefaultValue As Integer = 5
        Dim defaultReference As Object
```

توضح هذه السطور ثلاث متحولات منفصلة : اثنان من الأنواع ذات القيمة (النوع الصحيحIntegers)و النوع المرجعي (الكائنObject). على الرغم من أن متحول واحد فقط يملك إسناد بيانات صريح ،فإن جميع المتحولات الثلاث السابقة تم إسنادها بالفعل لشيء ما ، إما بشكل صريح أو ضمني . لننظر إلى هذه العبارات مرة أخرى و نرى ما تم إسناده بشكل صحيح إلى المتغير.

```
Dim defaultValue As Integer = 0
```

```
        Dim nonDefaultValue As Integer = 5
        Dim defaultReference As Object = Nothing
```

حدث كل من التصريح والإسناد مسبقاً لجميع المتحولات ، وذلك فقط باستخدام العبارة Dim . المتغير defaultValue ، مع إسناده الافتراضي إلى الصفر، يمكن أن يستخدم مباشرة في معادلات . على أية حال ،إن متحول النوع المرجعي defaultReference هو كوب شاي فارغ تماماً ، مع بيانات غير افتراضية لتتم معالجتها . يوجد ميزات في الفيجوال بيسك والتي تسمح لك أن تقارن الأنواع المرجعية مع لاشيء Nothing ، ويمكنك أن تفعل هذا مباشرة ، ولكنها بالواقع ليست بيانات .و تذكر ، إن المتغيرات تعيش لتدير البيانات .
تحتاج قيم الأنواع المرجعية للاستنساخ ، ويحتاج الاستنساخ لكلمة محجوزة جديدة New:

```
Dim defaultReference As Object = New Object
```

تشير الآن defaultReference إلى كائن حقيقي : يملك الآن كوب شاي defaultReference شيء ما قابل للاستهلاك داخله ، ورغم من أنه مجرد System.Object ، فهو لا يملك الكثير في الطرق. النصوص هامة أكثر بقليل ، وتملك أيضاً المشيدات الهامة.
المشيد هو مقطع من كود التمهيد initialization الذي ينفذ عندما تنشئ كائن أو نوع بيانات جديد . تسمح لك بعض الكائنات بأن تمرر معلومات إضافية إلى مشيد ما،والمعلومات الإضافية التي تم استخدامها في المعالجة التمهيدية . لايسمح لك المشيد الافتراضي*default constructor* بتوفير أية معلومات إضافية : إنه يعمل بنفسه فقط، مسنداً قيم أولية للبيانات بدون تدخل خارجي . لا يوجد حدود لعدد المشيدات في الفئة ، ولكن يجب على كل واحد أن ينوع في نوع المعلومات الإضافية الممررة إليه . لذلك فبالعودة إلى السلاسل الحرفية Strings. يعمل المشيد الافتراضي للسلسلة الحرفية ببساطة على إنشاء لوح فارغ blank ، أو سلسلة حرفية ذات طول صفري zerolength:

```
Dim worldsMostBoringString As String = New String
```

الآن، لا أحد يعمل هذا ، بما أن العبارة التالية تعمل بشكل جيد تماماً :

```
Dim worldsMostBoringString As String = ""
```

وذلك لأن السلاسل الحرفية Stringsتتم معالجتها بشكل خاص من قبل الفيجوال بيسك . إن المحارف النصية String literalsعملياً هي استنساخات من قيم البيانات النصية String: وكأنك أنشأت حالة (نسخة) نصية String جديدة باستخدام فئة System.String. ذلك صحيح على الأقل عند استخدام المشيد الافتراضي لأنواع البيانات النصية . ولكن تملك السلاسل الحرفية مشيدات هامة أكثر . (سأبحث في تفاصيل المشيدات في الفصل الثامن) . يعمل واحد من المشيدات على إنشاء حالة (نسخة) سلسلة نصية جديدة ممهدة بحروف خاصة مكررة عدد من المرات .
على سبيل المثال ، لإنشاء حالة نصية جديدة بسلسلة حرفية ذات 25 حرف من الحرف *M* . استخدم البناء التالي :

```
Dim mmGood As String = New String("M"c, 25)
```

إذا كنت ستستخدم نفس نوع البيانات فقط بعد الكلمة المحجوزة As بحيث تستخدم بعدها مباشرة الكلمة المحجوزة New، وبالتالي فإنك تستطيع استخدام التركيب المختصر(أو المطوي) كمايلي :

```
Dim mmGood As New String("M"c, 25)
```

كما مع الأنواع ذات القيمة ، يمكنك تقسيم العبارة إلى عبارة تصريح واضحة وعبارة إسناد :

```
    Dim mmGood As String
        mmGood = New String("M"c, 25)
```

الثوابت Constants

لا تتغير المحارف ، ولكنك تستطيع أن تستخدمها مرة فقط في كودك .إن الثوابت *Constants* هي المعبر بين المحارف و المتغيرات : ولديها قيمة مفردة غير قابلة للتغيير never-changing تماماً مثل محارف البيانات ، ولديها أيضاً اسم يمكنك أن تستخدمه بشكل متكرر، تماماً مثل المتحولات .
صرح عن الثوابت مستخدماً الكلمة المحجوزة Const بدلاً من الكلمة المحجوزة Dim :

```
Const SpeedOfLight As Integer = 186000
```

يحدث الإسناد الفعلي للقيمة إلى الثابت في العبارة نفسها ، مع القيمة التي تتبع المعامل = . حالما يتم التصريح والإسناد للثابت،فإنه يصبح متاح للاستخدام في عبارات فعلية من كودك الفعلي :

```
MsgBox("Lightspeed in miles/second: " & SpeedOfLight)
```

الحقول و التصريحات المحلية Local Declaration and Fields

في العالم الحقيقي ، تحتاج لأن تحفظ بعض كخاصة ، من أجل استخدامك فقط . ولكن توجد معلومات عامة ليست مخبأة عن أي شخص . وهذه الطريقة ليست فقط في العالم الواقعي : يملك العالم الزائف للفيجوال بيسك مستويات مختلفة من الوصول والخصوصية (السرية) لبياناتك . بعد قليل سوف نرى أن كود منطق تطبيقك سيظهر دائماً في إجراء ، مسمى مقطع الكود المصدري . إنك تصرح عن متحولات محلية*local variables* (و ثوابت constants) في نفس هذه الإجرائيات عندما تحتاج إلى متغير شخصي وقصير العمر short-livedوذلك فقط من أجل الاستخدام ضمن الإجراء الواحد . يمكن أن تظهر المتحولات الأخرى (و الثوابت) خارج الإجراء ،و لكن يبقى ضمن سياق فئة ما أو نوع مشابه . وهذه الحقول *fields*، أياً كانت متغيرات أو ثوابت ، هي متاحة مباشرة لجميع الإجراءات المختلفة والتي تستدعي أيضاً الفئة الحالية.
إنك تعرف جميع المتحولات المحلية مستخدماً الكلمة المحجوزة Dim . تعمل العبارة Dim من أجل تعريفات الحقول ، ولكن من الشائع أكثر استخدام كلمات محجوزة لمحددات وصول خاصة بدلاً منها . و تحدد هذه المعدلات(أو المحددات) أي كود يستطيع الوصول إلى الحقول ، من Private (مستخدماً فقط من قبل الكود داخل الفئة) إلى Public (أيضاً متغير خارج الفئة) :

```
Private ForInClassUseOnly As Integer
```

توجد خمس محددات وصول. سوف أناقشها أكثر وأناقش الحقول بشكل عام في الفصل السادس.

التعليقات Comments

إن اللغة الأجنبية في هذا الكتاب هي الفيجوال بيسك ، واللغة الإنكليزية هي اللغة العامية ، يمكنك أن تجد مقطع خاص من كود الفيجوال بيسك تمت كتابته على نحو رديء أو حتى مكروه ، ولكن إذا كانت التعليقات المصاحبة مضبوطة (دقيقة) ، مع اللغة البشرية human-languageالمفهومة للمعالجة .
تظهر التعليقات Commentsعادة على السطور بحد ذاتها ، ولكنك تستطيع أيضاً تلحق تعليق إلى نهاية سطر الكود الموجود . إذا تم قطع سطر منطقي ما إلى سطور حقيقية متعددة باستخدام حرف متابعة السطر "_" ، فإن تعليق مذيل يكون قاطعاً عند نهاية السطر الحقيقي النهائي :

```
' ----- This is a standalone comment, on a line by itself.
        Dim counter As Integer ' This is a trailing comment.
MsgBox("The counter starts at " & _ ' INVALID COMMENT HERE!
counter) ' But this one is valid.
```

تبدأ التعليقات مع حرف التعليق ، حرف علامة الاقتباس المفرد القياسي (‘). أي نص متبوع بحرف التعليق هو تعليق ، و يتم تجاهله عندما يتم ترجمة الكود داخل التطبيق القابل للاستخدام . يظهر أي مقطع وحيد ضمن نص محرفي لم يتم استخدامه كعلامة تعليق .

```
MsgBox("No 'comments' in this text.")
```

يمكن أيضاً أن تنشأ التعليقات بالكلمة المحجوزة REM (كما في "REMark ")، ولكن يستخدم معظم المبرمجين العلامات المفردة بدلاً منها .

العبارات الاختيارية Option Statements

بعض أمثلة الكود التي رأيتها سابقاً والتي يدعمها الفيجوال بيسك بقيمة إسناد افتراضية default assignment لمتغير ما ـ على الأقل من أجل الأنواع ذات القيمة ـ إذا أهملت تضمين واحد . في حالات محددة ، سوف يدعم الفيجوال بيسك أيضاً التصريحات *declaration*إذا تركتها خارجاً . في العبارة :

```
brandNewValue = 5
```

إذا لم يكن هناك عبارة Dim والتي تعرف brandNewValue ، فإن الفيجوال بيسك سوف يصرح عن المتغير بالنيابة عنك ، مسنداً إياه إلى نوع البيانات Object. لا تدع هذا يحدث لك ! فإنك لا تعرف أي نوع من المشاكل ستحصل إذا سمحت لمثل هذه الممارسات في كودك . سوف تجد بسرعة أن كودك مملوء بالعلل المنطقية الغامضة ، ونتائج بيانات خفية ، و هكذا .

المشكلة هي أن الفيجوال بيسك لن يحتج(يشتكي) إذا أخطأت بكتابة الاسم لمتغيرك المصرح عنه ذاتياً auto-declared،المتروك بدون اختبار ، مثل هذه الممارسات يمكن أن تؤدي إلى كود مثل هذا :

```
    brandNewValue = 5
            MsgBox(brandNewVlaue
```

يترجم فيجوال بيسك العبارة بدون أي أخطاء ؟ و الآن يعرض صندوق الرسالة لاشيء بدلاً من الرقم 5 ؟ بإمكانك أن تتجنب مثل هذه الصدمة بحكمة باستخدام العبارات Option الداخلة في لغة الفيجوال بيسك . ويوجد أربع أنواع لهذه العبارات :

- خيار التصريح Option Explicit On :

تجبرك هذه العبارة أن تصرح عن جميع المتغيرات مستخدماً Dim (أو عبارة مشابهة) قبل الاستخدام . من الممكن أن تستبدل مكان”On“ ب ”Off“ في العبارة ، ولكن لا تفعل هذا .

- خيار التدقيقOption Strict On :

سوف تعمل الفيجوال بيسك بعض تبديلات للبيانات البسيطة من أجلك عند الحاجة . على سبيل المثال ،إذا أسندت قيمة بيانات طويلة Long 64 ـ بت لمتغير صحيح Integer32 ـ بت ،عندئذ سوف تحول الفيجوال بيسك بشكل عادي هذه البيانات إلى حجم أصغر ، محتجة فقط إذا لم تتلاءم البيانات. هذا النوع من التحويل ـ تحويل تضيق ـ ليس آمناً بما أن البيانات المصدرية سوف تفشل أحياناً لتتناسب في المكان المحدد .

(تحويل التوسيع ، كما مع تخزين البيانات الصحيحة Integerفي الصحيح الطويل Long ، تعمل دائماً ، بما أن المكان المحدد دائماً يحجز القيمة المصدرية .) إن عبارة خيار التدقيق Option Strict On تتخلص من المعالجة الذاتية لتحويلات التضييق . و يتم إجبارك على أن تستخدم دوال التصريح explicit لتنجز تحويلات التضييق . وهذا جيد ، بما أنها أجبرتك أن تفكر حول نوع البيانات لمتغيراتك التي حجزتها . يمكنك أن تستبدل ”On“ ب ”Off“ في هذه العبارة ، ولكن إذا حذرتك مرة فإنني أحذرك مرتين :بأن لاتعمل هذا .

- خيار الاستنتاج Option Infer On :

تخبر عبارة الفيجوال بيسك 2008 الجديدة المترجم ليضع في حسبانه أي نوع من البيانات التي تريد من متحول ما أن يستخدمها خاصة عندما لا تخبره . سأناقش خيار الاستنتاج في الفصل السادس ، لذلك سوف أؤخر التفاصيل الآن . عادةً هذا الخيار فعال ”.On“

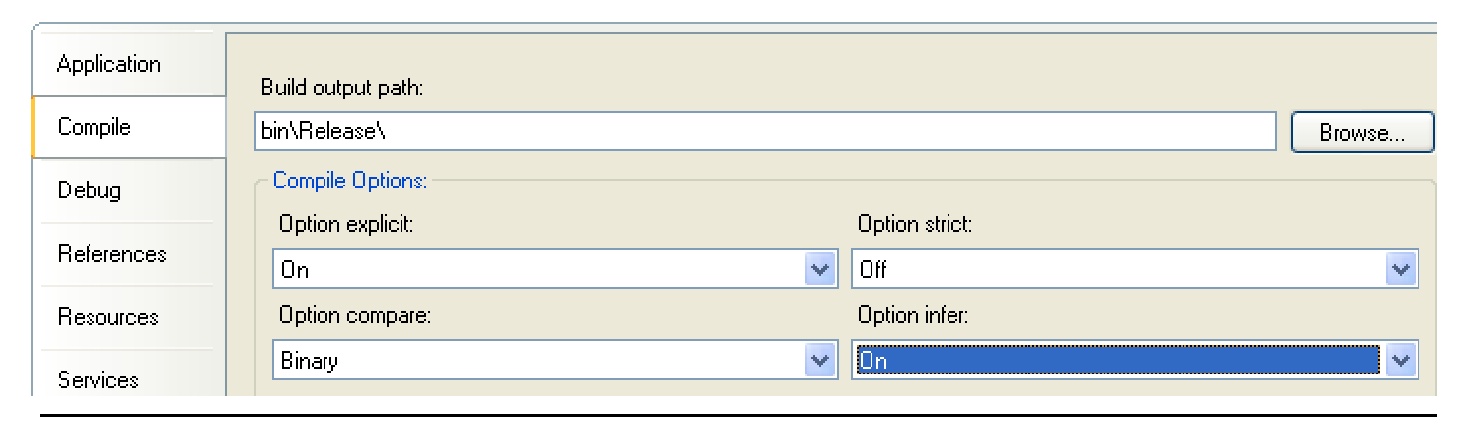
- خيار المقارنة الثنائي والنصي Option Compare Binary *and* Option Compare Text:

يوجه هذان التشكيلين من عبارة خيار المقارنة Option Compare كودك ليستخدم قواعد الفرز الخاصة من أجل ميزات مقارنة سلسلة نصية محددة . على العموم ، المقارنات الثنائية Binary هي حساسة لحالة الحروف case-sensitive ، حيث أن المقارنات النصية Text ليست كذلك . يعود الأمر إليك في أي طريقة تريد أن تستخدم :القيمة الافتراضية هي الثنائية . تظهر هذه العبارات عند أعلى كل ملف كود مصدري في مشروعك ، قبل أي كود آخر :

```
Option Explicit On
Option Strict On
```

أو لتوفير سعة على قرصك الثمين ،هناك القيم الافتراضية التي تقدمها لكامل مشروعك من خلال خصائص المشروع . في الفيجوال استديو ، اختر القائمة مشروع Project ثم أمر الخصائص Properties. ثم على نافذة خصائص المشروع التي تظهر ، اختر تبويب Compile و هيئ الاختيارات الافتراضية من أجل الخيارات التالية :

“Option explicit، ”Option strict,“ ،”Option compare و“ ”Option infer“ (انظر الشكل التالي):



معاملات أساسية Basic Operators

تتضمن الفيجوال بيسك عمليات أساسية عديدة و التي تدعك تعمل ما يريد أن يفعله كودك فعلاً : معالجة البيانات . تتيح لك معاملات الفيجوال بيسك أن تنجز دوال إدارة رياضية ، منطقية ،عددية ثنائية ، ونصية ، وجميعها بدون كلفة إضافية.

المعامل الأساسي هو معامل الإسناد ، ممثل بواسطة إشارة المساواة (=) . لقد رأيت مسبقاً هذا المعامل في هذا القسم . استخدمه لتسند بعض القيم إلى متغير ما (أو ثابت) : يتم إسناد كل ما يظهر على يمين المعامل إلى متغير النوع المرجعي أو النوع ذي القيمة الذي على اليسار. العبارة :

```
fiveSquared = 25
```

هي إسناد قيمة 25 إلى المتغير fiveSquared . معظم المعاملات هي معاملات ثنائية *binary operators* ـ وهي تعمل على قيمتين متميزتين ، واحدة على يسار المعامل و واحدة إلى اليمين : النتيجة هي قيمة محسوبة وحيدة .وكما مع الحسابات يتم استبدالها بشكل كامل بالنتيجة المحسوبة . على سبيل المثال ، عملية الجمع :

```
seven = 3 + 4
```

تصبح

```
seven = 7
```

قبل التطبيق النهائي لمعامل الإسناد (=) . يظهر المعامل الفردي unary operator إلى اليسار فقط من معامله . على سبيل المثال ، معامل النفي الأحادي يحول العدد الموجب إلى عدد سالب :

```
negativeSeven = -7
```

يضع الجدول التالي قائمة معاملات الفيجوال بيسك الرئيسية و يصف باختصار الغرض من كل واحدة .

المعامل	الوصف
+	يضيف معامل الجمع عددين معاً .
+	يحفظ معامل الزائد الفردي الإشارة لقيمة عدد ما . و هو ليس مفيد جداً لكي تحصل على معامل تحميل زائد
-	يطرح معامل الطرح العامل الثاني من الأول .
-	يعكس معامل النفي الفردي الإشارة لعامله العددي المقرون به
*	يضرب معامل الضرب قيمتين عدديتين ببعضهما .
/	يقسم معامل القسمة العامل العددي الأول بواسطة الثاني ، مرجعاً ناتج القسمة المتضمنة أي باقي عشري.
\	يقسم معامل قسمة الأعداد الصحيحة العامل العددي الأول بواسطة الثاني ، مرجعاً ناتج القسمة ،مع قطع الباقي العشري .
Mod	يقسم معامل الباقي العامل العددي الأول بواسطة الثاني ، و يعود بالباقي فقط كقيمة صحيحة .
^	يرفع المعامل الأسي العامل الأول (الأساس) إلى قوة الثاني (الأس) .
&	يربط معامل توحيد السلاسل الحرفية عاملي سلسلتين حرفيتين مع بعضهما ، و يحولها إلى سلسلة جديدة بنتائج مترابطة .
And	يعود معامل الربط هذا بقيمة صح True إذا كانت كل العوامل المنطقية أيضاً صح True.
AndAlso	يشبه هذا المعامل تماماً معامل And ، و لكنه لا يفحص أو يعالج العامل الثاني إذا كان الأول خطأ False .
Or	يعود معامل الربط هذا بقيمة صح True إذا كان إحدى العاملين أيضاً صح True.
OrElse	يشبه هذا المعامل تماماً معامل Or، ولكنه لا يفحص أو يعالج العامل الثاني إذا كان العامل الأول صح True.
Not	يعود معامل النفي بعكس العامل المنطق.
Xor	يعود معامل"أو" الخاص بالقيمة صح True إذا كان واحد وواحد فقط من العوامل صح True.
<<	يرفع معامل الرفع الأيسر البتات المستقلة في عامل عددي صحيح ما إلى اليسار باستخدام عدد البتات في العامل الثاني .
>>	يرفع معامل الرفع الأيمن البتات المستقلة في معامل عددي صحيح ما إلى اليمين باستخدام عدد مواقع البتات في العامل الثاني .
=	يعود معامل المقارنة يساوي إلى equal-to القيمة صح True إذا كانت المعاملات مساوية "equal" بعضها البعض .
<	يعود معامل المقارنة أقل من less-than القيمة صح True إذا كان العامل الأول أقل من "less than" الثاني .
<=	يعود معامل المقارنة أقل أو يساوي إلى less-than-or-equal-to القيمة صح True إذا كان العامل الأول أقل أو يساوي العامل الثاني "less than or equal to" .
>	يعود معامل المقارنة أكبر من greater-than القيمة صح True إذا كان العامل الأول أكبر من الثاني "greater than" .
>=	يعود معامل المقارنة أكبر من أو يساوي إلى greater-than-or-equal-to القيمة صح True إذا كان العامل الأول أكبر من أو يساوي العامل الثاني "greater than or equal to" .
<>	يعود معامل المقارنة لا يساوي إلى not-equal-to القيمة صح True إذا كان العامل الأول لا يساوي إلى الثاني "not equal to" .
Like	يعود معامل نموذج المقارنة القيمة صح إذا طابق العامل الأول نموذج سلسلة حرفية محددة من قبل العامل الثاني.
Is	يعود معامل مقارنة كائن يساوي إلى ـ القيمة صح إذا كان كلا العاملين يمثلان نفس الحالة لنوع بيانات ما في الذاكرة .

	وضع العامل الثاني إلى القيمة لاشيء Nothing تدعك تختبر المتغير المرجعي لترى فيما إذا يحتوي البيانات أم لا .
IsNot	معامل مقارنة كائن لا يساوي إلى not-equal-to هو عكس المعامل Is.

بقدر فعالية المعاملات،فهي تستحوذ على فعالية أكبر عندما تعمل على دمجها.وهذا يعمل لأن أياً من العوامل يمكن أن تكون تعابير معقدة بحيث تتضمن عوامل خاصة بها.الأقواس المضمنة حول شروط في عامل تضمن معالجة القيم في الترتيب الذي تتوقعه.

```
circleArea = pi * (radius ^ 2)
```

في هذه العبارة ، العامل الثاني لمعامل الضرب * هو تعبير آخر ، و الذي يتضمن المعامل الخاص به .

استخدام الدوال و الإجراءات الفرعية Using Functions and Subroutines

تسمح لك اللغات الإجرائية أن تقسم كودك داخل ما يسمى مقطع منطقي ، تدعى الإجراءات . تتيح هذه الإجراءات " التقسيم و التحكم " بالوصول إلى المحاكاة البرمجة: وأنت تكتب الإجراءات التي تنجز قسم منطقي محدد من الكود ضمن تطبيقك الكامل ، و بعدئذ الوصول إلى هذه الإجراءات من الإجراءات الأخرى . تتضمن الفيجوال بيسك Visual Basic ثلاثة أنواع من الإجراءات وهي:

- الروتينات الفرعية *Subroutines*:
  هذه الإجراءات ، تدعى أيضاً الإجراءات الفرعية *subprocedures* ، اعمل إجراء جزئي و بعدئذ عد إلى استدعاء الإجراء . يمكن أن يتم إرسال البيانات إلى داخل الإجراء الجزئي من خلال قائمة المعاملات النسبية *argument list* . و يمكن أن تعود بعض القيم من خلال نفس تلك القائمة ، ولكن لا يرسل الإجراء نتيجة نهائية رسمية للخلف . يعمل الإجراء الجزئي عمله ، وحالما يكتمل ، يستمر استدعاء الكود على طريقته النشيطة .
- الدوال *Functions* :
  تشبه الدوال الإجراءات الجزئية تماماً ، مع ميزة إضافية واحدة : وهي أنك تستطيع أن تعيد قيمة وحيدة أو حالة كائن من الدالة كنتيجته الرسمية . وبالعادة ، يأخذ الاستدعاء هذه القيمة الراجعة بعين الاعتبار عندما يكتمل منطقه الخاص.
- الخاصيات *Properties* :
  عندما استعملت ،الخاصيات والتي فعلاً تبدو كالمتغيرات . أسندت و استرجعت قيماً إلى و من الخاصيات تماماً كما عملت مع المتغيرات . على أية حال ، تتضمن الخاصيات الكود المخفي ، و المستخدم غالباً لتثبيت البيانات كونها مسندة إلى الخاصية .

الإجراءات الفرعية و الدوال و الخاصيات هي أعضاء الكود لكل فئة أو نوع مشابه . سوف أؤخر نقاش الخاصيات قليلاً فيما بعد في القسم . أما الآن، دعنا نستمتع بالدوال و الإجراءات ، اللذان يعرفان مع بعضهما البعض بالطرق *methods*. ولنبدأ بالإجراءات الفرعية . لكي تستدعي إجراء فرعي ما، اكتب اسمه كجملة ، متبوعة بمجموعة قوسين . أي بيانات تحتاج لكي ترسلها إلى الإجراء تذهب في الأقواس . على سبيل المثال ، يعمل استدعاء الإجراء الفرعي التالي بعض العمل ، ممرراً العدد ID للزبون ، و `startDate`:

```
DoSomeWork(customerID, startDate)
```

يعرف كل إجراء فرعي نوع البيانات و ترتيب المعاملات النسبية التي ممررتها . تضع هذه المعاملات النسبية قائمة يمكن أن تتضمن واحد أو أكثر من المعاملات النسبية الاختيارية *optional arguments* ، التي يتم إسنادها إلى القيم الافتراضية إذا لم تتضمنها . من الممكن أن يكون الإجراء الفرعي معاد تعريفه *overloaded*. ومعرفاً قوائم معاملات نسبية ممكنة مختلفة مبنية على العدد ونوع بيانات المعاملات النسبية . سوف نصادف الكثير من هذه فيما بعد .

الدوال أكثر متعة قليلاً بما أنها تعيد القيمة القابلة للاستخدام . وغالباً ، يتم إسناد هذه القيمة إلى متغير ما :

```
Dim balanceDue As Boolean
        balanceDue = HasOutstandingBalance(customerID)
```

وبعدئذ باستطاعتك عمل شيء ما بهذه النتيجة . إذا أردت ، يمكنك أن تتجاهل القيمة العائدة لدالة ما ، و تكون لدينا مسبقاً . تعيد الدالة المستعملة باكراً MsgBox المطابقة من زر منقور على الشاشة on-screen من قبل المستخدم لإغلاق صندوق الرسالة .إذا ضمنت فقط الكلمة OKفي الزر (القيمة الافتراضية)، فمن المحتمل أنك لا تهتم بأي زر ينقر المستخدم .

```
MsgBox("Go ahead, click the OK button.")
```

ولكن بإمكانك أيضاً أن تتحكم بالنتيجة من الزر :

```
whichButton = MsgBox("Click Yes or No.", MsgBoxStyle.YesNo)
```

في هذه الحالة ، whichButton سوف يكون إما MsgBoxResult.Yes أو MsgBoxResult.No ، لقد تم تعريف النتيجتين الممكنتين من قبل الدالة MsgBox.

الشروط Conditions

في بعض الأوقات يتوجب عليك أن تصنع بعض الاختيارات ، و سوف تساعدك التعابير الشرطية أن تعمل ذلك تماماً . تتضمن الفيجوال بيسك دعماً للشروط ، التي تستخدم اختبارات البيانات لتحدد الكود التالي الذي يجب أن تتم معالجته .

عبارات الشرط If

العبارة الشرطية الأكثر شيوعاً هي عبارة If . وهي مكافئة للأسئلة الإنكليزية في الصيغة " إذا كان كذا و كذا صحيحاً ، عندئذ نفذ كذا و كذا . " على سبيل المثال ، يمكنها أن تعالج " إذا كان لديك $20 ، عندئذ يمكنك أن تشتري لي طعام الغداء ، " تملك عبارة If قواعد لغوية والتي تقيس تعدد سطور الكود المصدري :

```
1:              If (hadAHammer = True) Then
2:                  DoHammer(inTheMorning, allOverThisLand)
3:                  DoHammer(inTheEvening, allOverThisLand)
4:              ElseIf (hadAShovel = True) Then
5:                  DoShovel(inTheNoontime, allOverThisLand)
6:              Else
7:                  TakeNap(allDayLong, onMySofa)
8:              End If
```

تتيح لك عبارة If تعريف التفرعات في كودك المبني على الشروط . إنه يبنى من ثلاثة عناصر رئيسية :وهي .

- الشروط *Conditions* :

إن التعبير الموجود بين الكلمة If (أو ElseIf) و الكلمة المحجوزة Then هو الشرط . يتضمن الاختبار شرطين ، على السطرين 1 و 4 . ومن الممكن أن تكون الشروط بسيطة أو معقدة ، ولكن يجب أن تنتج دائماً بقيم منطقية Boolean صحيحة True أو خاطئة False.و يمكن تتضمن استدعاءات لدوال أخرى و معاملات مقارنة و منطقية متعددة .

```
If ((PlayersOnTeam(homeTeam) >= 9) And(PlayersOnTeam(visitingTeam) >= 9)) Or _
        (justPracticing = True) Then
            PlayBall()
        Else
            StadiumLights(turnOff)
        End If
```

يتبع الشرط الأصلي دائماً الكلمة المحجوزة If. إذا فشلت تلك الشروط، تستطيع أن تحدد شروط إضافية تتبع الكلمة المحجوزة ElseIf، كما على السطر 4 . يمكنك أن تكتب عدد من عبارات ElseIf كما تحتاج . لا يتيح لك الشرط الاختياري Else أن تحدد تعبير اختبار ما. بدلاً من ذلك، إنه يصل كل شيء لم يتم ضبطه بعد بواسطة عبارات If أو ElseIf. يتم السماح بعبارة Elseواحدة فقط لكل عبارة If .

- التفرعات *Branches* :

تكون كل كلمة شرط محجوزة *Then* متبوعة بواحد أو أكثر من عبارات الفيجوال بيسك التي يتم معالجتها إذا تم تقييم الشروط المرافقة إلى صح True. جميع العبارات حتى تصل إلى Else، ElseIf،أو End If التالية تكون مضمنة في مقطع عبارة التفرع(أي تتم معالجتها حتى تصل إلى التفرع التالي) . يمكنك أن تضمن أي عدد من العبارات في قطعة تفرع ما ، وحتى عبارات If ثانوية إضافية . في الكود البسيط ،تتم معالجة سطور التفرع 2 و 3 إذا كان الشرط الأصلي hadAHammer صحيحاً . تتم معالجة السطر 5 بدلاً من ذلك إذا سقط الشرط الأصلي ، ولكن إذا ما مر الشرط الثاني hadAShovel .إذا لم يكن أي من الشروط صحيحاً True ، فإن تفرع Else ، على السطر 7 ، ينفذ .

- الكلمات المحجوزة *Statement keywords*:

إن عبارة Ifهي واحدة من العديد من عبارات الفيجوال بيسك المتعددة الأسطر ، وجميعها تنتهي بالكلمة المحجوزة End متبوعة باسم الكلمة المحجوزة الأصلية (If في هذه الحالة) . تتضمن الكلمات المحجوزة لعبارة If، والتي تمنح العبارة تركيبها : If ، Then ، ElseIf ، Else، و End If . إن جميع عبارات Elseو ElseIf والفروع ذات العلاقة هي اختيارية . تتضمن عبارة If الأبسط تفرع If فقط.

```
If (phoneNumberLength = 10) Then
            DialNumber(phoneNumber)
        End If
```

من أجل الشروط ذات التفرعات وبعبارة مفردة بسيطة وبدون شروط ElseIf،فسطر واحد بديل يمكن أن يحفظ كودك أوضح.

```
If (SaveData() = True) Then MsgBox("Data saved.")
        If (TimeOfDay >= #1:00:00 PM#) Then currentStatus = WorkStatus.GoHome Else currentStatus =
WorkStatus.BusyWorking
```

إن عبارات If جامدة لأنها تجعل كودك أكثر من مجرد تعليمات مجموعة خطية مملة تدريجياً والتي تنحرف لأي سبب . تتم كتابة البرمجيات لتدعم بعض عمليات العالم الحقيقي . و عمليات العالم الحقيقي نادراً ما تكون خطية . تجعل عبارة If إمكانية كودك ليستجيب إلى مختلف شروط البيانات ، آخذاً التفرع المناسب عند الضرورة .

حالما تدخل If...End If تكتمل القطعة ، وتستمر المعالجة بالعبارات التالية التي تلي العبارة End If.

عبارات اختيار حالة Select Case Statements

بعض الأحيان من المحتمل أن تكتب عبارة If والتي تختبر متغير على قيمة محتملة وحيدة،مع thenأخرى،وthen ، و then،وهكذا:

```
    If (billValue = 1) Then
            presidentName = "Washington"
        ElseIf (billValue = 2) Then
            presidentName = "Jefferson"
        ElseIf (billValue = 5) Then
            presidentName = "Lincoln"
...
```

وعليه تذهب ، خلال عبارات ElseIf كثيرة متعددة . إنها فعالة ، ولكن مملة قليلاً ، كما أن على كودك أن يختبر بشكل خاص كل حالة . تدعم عبارة اختر حالة Select Case بديل أوضح من أجل مقارنات قيمة بسيطة على قائمة ما :

```
1:      Select Case billValue
02          Case 1
3:              presidentName = "Washington"
4:          Case 2
5:              presidentName = "Jefferson"
6:          Case 5
7:              presidentName = "Lincoln"
8:          Case 20
9:              presidentName = "Jackson"
```

```
10:          Case 50
11:              presidentName = "Grant"
12:          Case 10, 100
13:              presidentName = "!! Non-president"
14:          Case Is > 100
15:              presidentName = "!! Value too large"
16:          Case Else
17:              presidentName = "!! Invalid value"
18:          End Select
```

على خلاف عبارة If، التي تعمل اختبار من أجل نتيجة منطقية ما ،تقارن Select Case قيمة مفردة مع مجموعة ما من قيم حالة اختبار ما . في المثال ، تتم مقارنة المتغير billValue مع القيم المختلفة المعرفة بواسطة كل عبارة حالة Case. جميع الكود الذي يلي عبارة حالة ما Case(حتى عبارة الحالة التاليةCase) هو التفرع الذي تتم معالجته عندما تتم عملية المقارنة. يمسك شرط اختياري Case Else(السطر 16) أي شيء لا يمكن أن تتم مطابقته مع أي حالة Case أخرى . و عادة ، عباراتCase تجدول قيم وحيدة من أجل المقارنة . بإمكانها أيضاً أن تتضمن قائمة ما من قيم مقارنة الفواصل المنفصلة (السطر 12) ، أو تعابير مقارنة مجال بسيط (السطر 14) .

IIf وIf (IIf and If Functions)

تتضمن الفيجوال بيسك اختلافين لعبارات If من أجل الاستخدام المتوازي . خذ العبارة التالية :

```
If (gender = "F") Then fullGender = "Female" Else fullGender = "Male"
```

باستخدام الدالة IIf، يتم ضغط هذه العبارة في عبارة إسناد وحيدة مع شرط مدمج :

```
fullGender = IIf(gender = "F", "Female", "Male")
```

تملك الدالة IIf ثلاث معاملات نسبية محدودة بالفواصل comma-delimited . المعامل الأول هو الشرط ، الذي عليه أن يظهر في قيم منطقية Booleanإما صح Trueأو خطأFalse .المعامل الثاني الذي يتم إعادته من قبل الدالة إذا كانت قيمة الشرط صح True: تعيد نتيجة الشرط الخاطئة العامل النسبي الثالث . من أجل الشروط البسيط المخصصة لإعادة القيم إلى المتغيرات المشتركة ، إنه حقاً دالة مفيد . ولكن مع أي شيء مفيد حقاً ، يوجد ثلاث تحذيرات . التحذير مع IIf هو أن أي شيء يظهر داخل العبارة IIf ستتم معالجته ، حتى لو لم تتم إعادته كنتيجة . ولدينا هنا مثال خطير :

```
purgeResult = IIf(level = 1, PurgeSet1( ), PurgeSet2( ))
```

ستعيد العبارة بشكل صحيح النتيجة من كل من PurgeSet1( ) أو PurgeSet2( )المبنية على قيمة المستوى . إن المشكلة ، أو المشكلة المحتملة ، هي أن كلاً من الدوال ، PurgeSet1( ) ، PurgeSet2() سوف يتم استدعاءها : إ ذا كان المستوى هو 1 ، إن كل من PurgeSet1( ) و PurgeSet2( ) سوف يتم استدعاءه ، على الرغم من أن النتيجة من الدالة PurgeSet1()فقط ستتم إعادتها .

للمساعدة على تجنب مثل هذه التأثيرات الجانبية ، أضافت الفيجوال بيسك 2008 معامل If جديد . إنه يبدو تماماً مثل الدالة IIf، ما عدا من أجل الكلمة المحجوزة If التي تحل محل الكلمة المحجوزة IIf:

```
ClasspurgeResult = If(level = 1, PurgeSet1( ), PurgeSet2( ))
```

سوف يتم الآن استدعاء PurgeSet1( )أو PurgeSet2( ) فقط بناء على الشرط ، و لكن ليس كلاهما . على الرغم من أن المعامل If يبدو كدالة ،وهو فعلاً معامل صحيح ، معروف بمعامل ثلاثي *ternary*. في وقت الترجمة ، تعالجه الفيجوال بيسك وتعالج معاملاته النسبية كمعاملات و عوامل وتولد المنطق المناسب . يأخذ تنوع للمعامل If معاملين نسبيين فقط ، باستثناء المعامل النسبي المنطقي Boolean الأولي .

```
realObject = If(object1, object2)
```

في هذا الإصدار للمعامل If، إذا أخذ المعامل النسبي الأول القيمة لا شيء Nothing ، فإن المعامل سيعيد المعامل النسبي الثاني . إذا لم يكن المعامل النسبي الأول لا شيء Nothing - بمعنى أنه ، إذا كان فعلاً شيء ما - فإن المعامل يعيد ذلك المعامل النسبي الأول عوضاً . الهدف عدم إعادة لاشيء non-Nothing .

الحلقات Loops

تتضمن الفيجوال بيسك ثلاث أنواع رئيسية من الحلقات وهي : For...Next و For Each...Next و Do...Loop . تسمح لك بتقسيم التكرار المتعاقب لكودك من خلال التفرعات ، تضيف الحلقات الفائدة لكودك حيث أنها تتيح لك أن تكرر قطعة محددة من منطق مثبت أو متغير عدد من المرات .

حلقات For...Next

تستخدم الحلقة For...Next عداد عددي والذي يزداد من قيمة بدائية إلى قيمة نهائية ، معالجاً الكود ضمن الحلقة حالاً من أجل كل قيمة متزايدة .

```
    Dim whichMonth As Integer
For whichMonth = 1 To 12
ProcessMonthlyData(whichMonth)
Next whichMonth
```

هذا نموذج يدور 12 مرة (من 1 إلى 12) ، مرة لكل شهر . يمكنك أن تحدد أي قيم للبداية والنهاية: يمكن أن يتم تحديد هذا المجال باستخدام متغيرات أو دوال تعيد قيم عددية . حالما يتم الحصول على قيم البداية والنهاية ، لن يتم إعادة حسابها كل مرة خلال الحلقة ، حتى لو تم استخدام استدعاء الدالة للحصول حد واحد أو كلا الحدين .

```
          ' ----- Month(Today) تعود بالشهر كعدد
          ' من أجل التاريخ الحا
          For whichMonth = 1 To Month(Today)
              ProcessMonthlyData(whichMonth)
          Next whichMonth
```

عادة ، تزداد الحلقة بواسطة (1) خلال كل مرة . يمكنك أن تبدل هذه القيمة الابتدائية بضم عبارة Step إلى نهاية سطر العبارة For:

```
For countDown = 60 To 0 Step -1
```

```
...
Next countDown
```

يسمح لك تباين قاعدي إضافي أن تصرح عن متغير عداد الحلقة ضمن العبارة ذاتها . مثل هذه المتغيرات متاحة ضمن الحلقة فقط ،ويتوقف وجودها عند خروجك من الحلقة .

```
        For whichMonth As Integer = 1 To 12
            ProcessMonthlyData(whichMonth)
        Next whichMonth
```

حلقات For Each. . .Next

الحلقة For Each...Next تطور عن الحلقة For، وتعمل هذه الحلقة بحث خلال مجموعة من البنود المرتبة والمترابطة ، من البند الأول حتى الأخير . وتعمل أيضاً على كائنات المصفوفاتArrays و التجمعاتcollection. كما يعمل أي كائن يدعم الواجهة IEnumerabl . إن التركيب مشابه تماماً للعبارة القياسية For :

```
        For Each oneRecord In setOfRecords
            ProcessRecord(oneRecord)
        Next oneRecord
```

حلقات Do. . .Loop

تريد أحياناً أن تكرر قطعة من الكود طالما يكون الشرط المحدد صحيحاً ، أو فقط حتى يكون الشرط صحيحاً . إن التركيب Do...Loop ينجز كلاً من هذه المهام . تتضمن العبارة Do...Loop عبارة While أو Until التي تحدد الشروط من أجل المعالجة المستمرة للحلقة. على سبيل المثال ، تعمل العبارة التالية معالجة ما لمجموعة ما من التواريخ، من بداية تاريخ ما إلى نهاية تاريخ :

```
        Dim processDate As Date = #1/1/2000#
        Do While (processDate < #2/1/2000#)
            ' عمل معالجة للتاريخ الحالي
            ProcessContent(processDate)
            ' التقدم إلى البيانات التالية
            processDate = processDate.AddDays(1)
        Loop
```

سوف تستمر معالجة البيانات في هذا النموذج حتى تلاقي المتغير processDate أو تجتاز التاريخ 2/1/2000،الذي يشير إلى نهاية المعالجة . إن إصدار العبارة Until مشابه لما تقدم ، ولكن نتيجة الشرط ستكون معكوسة :

```
Do Until (processDate >= #2/1/2000#)
...
        Loop
```

اجعل الشرط المضمن بسيطاً أو معقداً كما تريد . واضعاً العبارة Until أو Whileعند أسفل الحلقة يكفل أن العبارات التي في داخل الحلقة سوف تتم معالجتها دائماً مرة على الأقل:

```
Do
...
Loop Until (processDate >= #2/1/2000#)
```

إذا لم تتم مقابلة شرط الحلقة ، سوف تتابع الحلقة التنفيذ ولم ينتهي تنفيذها . لذلك إذا أردت أن تخرج حلقتك عند نقطة ما (و بالعادة أنت تفعل هذا) ، تأكد أن تتم مقابلة الشرط في النهاية . ويوجد حلقة أخرى مشابهة للحلقة Do...Loop ، تدعى الحلقة While...End While. وعلى أي حال ،فإنها توجد من أجل توافق تراجعي فقط . استخدم بدلاً منها العبارة Do...Loop.

عبارات "خروج" Exit Statements

عادة ، عندما تدخل حلقة ما ، فإنه لديك غرض من كل تكرار للعدد الكامل من المرات المحددة بالشروط الأولية للحلقة . من أجل حلقات For، فتوقع أن تستمر من خلال المجال العددي المدخل أو مجموعة العناصر . في حلقات Do ، فإنك تخطط لكي تحافظ على استمرار الحلقة طالما لم تقابل شرط الخروج حتى الآن . ولكن ربما هناك حلقات تريد أن تخرج منها باكراً . وتبلغ هذا باستخدام عبارة Exit.

ويوجد عبارتي Exit خاصتين بالحلقاتloop-specific : وهما :

- Exit For :
  تخرج من الحلقة For...Next أو الحلقة For Each...Next بشكل مباشر.
- Exit Do
  تخرج من عبارة Do...Loop مباشرة.

تخرج كل عبارة Exit من الحلقة التي تحوي العبارة : تستمر المعالجة مع السطر الذي يلي الحلقة مباشرة :

```
        For whichMonth = 1 To 12
            If (ProcessMonthlyData(whichMonth) = False) Then Exit For
        Next whichMonth
        ' يـتم وضع كـود الاستمرار هنا لامشكلة في كيفية الخروج من الحلقة
```

يتم تصميم نموذج الكود للحلقة من خلال جميع ال 12 شهر . على أية حال فشل المعالجة من أجل أي شهر من ال12 شهر سوف يغادر الحلقة مباشرة ، تاركاً جميع أعمال معالجة الشهر المتبقي . تخرج العبارة Exit Do بشكل مشابه من حلقات Do...Loop مباشرة.

في عبارة حلقة الخروج Exit ضمن حلقات متداخلة nested loops) حيث تظهر حلقة ما ضمن حلقة أخرى)، فإن الحلقة الموافقة فقط التي تحوي العبارة مباشرة يتم خروجها:

```
For whichMonth = 1 To 12
    For whichDay = 1 To DaysInMonth(whichMonth)
        If (ProcessDailyData(whichMonth, whichDay) = False) _
        Then Exit For
    Next whichDay
    ' عند الخروج من الحلقة الداخلية يأتي التنفيذ إلى هذا السطر
    ' متابعاً المعالجة من أجل الشهر التالي
Next whichMonth
```

عبارات الاستمرار Continue Statements

بما أن الخروج من حلقة ما يهجر جميع المسارات المتبقية خلال الحلقة ، فمن المحتمل أنك تتغاضى عن عملية معالجة بيانات هامة والتي سوف تحدث بطرق متتالية ، تتضمن الفيجوال بيسك عبارة Continueوالتي تتيح لك أن تترك المسار الحالي فقط خلال الحلقة . يوجد تنوع لعبارات Continueمختلفة من أجل كل نوع للحلقات :

- Continue For :
  تقفز مباشرة إلى نهاية الحلقة For...Next أو الحلقة For Each...Next و تتجهز للمسار التالي . تتم زيادة متغير الحلقة ومقارنته مع المجال أو حدود التجمع.
- Continue Do :
  تقفز مباشرة إلى نهاية العبارة Do...Loop وتتجهز من أجل المسار التالي . تتم إعادة تقييم الشرط Until أو الشرط While.

بما أنه تتم تقييم شروط الحلقة عند استخدام العبارات Continue، فإنه توجد مواعيد عندها يمكن أن تسبب Continue إغلاق الحلقة ، كما عندما يتم إنهاء المسار خلال الحلقة مسبقاً . في هذا المثال ، تقفز العبارة Continue For معالجة الشهور التي ليس لديها بيانات للمعالجة :

```
For whichMonth = 1 To 12
    If (DataAvailable(whichMonth) = False) Then Continue For
    RetrieveData(whichMonth)
    ProcessData(whichMonth)
    SaveData(whichMonth)
Next whichMonth
```

إنشاء إجراءاتك الخاصة Creating Your Own Procedures

يجب أن تظهر جميع العبارات المنطقية في كودك ضمن إجراء ما ، إما في إجراء فرعي ، دالة ما ، أو خاصية ما . على الرغم من أنه يوجد الآلاف من الإجراءات المسبقة الكتابة من أجلك كي تختار منها في مكتبات إطار عمل الدوت نت ، فإنه بإمكانك أن تضيف إجراءاتك الخاص أيضاً.

الإجراءات الفرعية Subroutines

تبدأ الإجراءات الفرعية بعبارة تصريح Sub ما و تنتهي بعبارة End Sub. ويظهر جميع منطق إجراءاتك الفرعية بين هذين الفكين الرائعين .

```
01 Sub ShowIngredients(ByVal gender As Char)
2:     Dim theMessage As String = "Unknown."
3:     If (gender = "M"c) Then
4:         theMessage = "Snips and snails and puppy dog tails."
5:     ElseIf (gender = "F"c) Then
6:         theMessage = "Sugar and spice and everything nice."
7:     End If
8:     MsgBox(theMessage)
09 End Sub
```

يظهر السطر 1 سطر تصريح الإجراء الفرعي بأبسط نماذجه : على طوال الكتاب ، سوف تجد أن هناك كلمات محجوزة إضافية والتي تجمل تصريحات الإجراء لتغير سلوكها . تبدأ العبارة مع الكلمة المحجوزة Sub (من أجل الإجراء الفرعي) ،متبوعاً باسم الإجراء ، ShowIngredients . تحوي الأقواس التالية لهذا الاسم وسيطات *parameters* الإجراء الفرعي . تسمح الوسيطات لمقطع آخر من الكود الذي سيستخدم هذا الإجراء أن يمرر البيانات ضمن الإجراء، وأن يتلقى بشكل اختياري البيانات العائدة . يمكنك أن تدخل أي عدد من الوسيطات في تعريف الإجراء الفرعي : بشكل بسيط افصلها بفواصل . يعيّن كل وسيط الاسم بأن يتم استخدامه في الإجراء (المتغير gender في النموذج) و نوع البيانات (محرفي Char) . تتم معالجة المعاملات النسبية كمتغيرات مصرح عنها ضمن الإجراء ، كما تم عمله مع المتغير genderعلى السطرين 3 و 5 .

القيم المزودة بواسطة الكود المستدعي تعرف بالمعاملات النسبية *arguments*. يتم تمرير جميع المعاملات النسبية بالقيمة *by value* أو بالمرجع *by reference*. في نموذج الكود ، سيتم تمرير المعامل النسبي الممرر إلى المتغير gender بالقيمة ، كما تم تعينها من خلال الكلمة المحجوزة ByVal. تشير الكلمة المحجوزة المرتبطة ByRef إلى معامل نسبي ما سيتم تمريره بالمرجع . وإذا لم تدخل أي كلمة محجوزة ، فإنه يتم فرض ByVal. تؤثر طريقة التمرير هذه فيما إذا حصلت تغييرات للمعامل النسبي ضمن الإجراء المحلي وتم إرسالها مرةً أخرى إلى كود الاستدعاء . على أية حال ، تتأثر القدرة على تحديث البيانات الأصلية أيضاً فيما إذا كانت البيانات من النوع ذو القيمة *value* type أو من النوع المرجعي *reference type*. و يشير الجدول التالي إلى السلوك المتبع عند دمج طريقة التمرير و نوع البيانات .

طريقة التمرير	نوع البيانات	السلوك
ByVal	Value type	التغيرات التي تحدث للإصدار المحلي من المعامل النسبي التي لا تملك التأثير على الإصدار الأصلي .
ByVal	Reference type	التغيرات التي تحدث على أعضاء *members* كائن البيانات تؤثر مباشرة على كائن البيانات الأصلي. على أية حال ، الكائن بحد ذاته لا يمكن أن يتم تغييره أو استبداله بكائن بيانات جديد بشكل

		كامل .
ByRef	Value type	التغيرات التي تحدث على الإصدار المحلي يتم إعادتها إلى الإجراء المستدعي ، وتؤثر دائماً على قيم البيانات الأصلية.
ByRef	Reference type	التغيرات التي تحدث سواء لكائن البيانات أو أعضائه تتغير أيضاً في الأصل . من الممكن أن تستبدل بشكل كامل الكائن المرسل إلى الإجراء .

في معظم الحالات ، إذا أردت أن تستمتع في تعديل القيمة لوسيط ما و امتلاك المتغيرات التي ترجع إلى المستدعي ، استخدم ByRef، وإلا ، فاستخدم ByVal. تشمل السطور من بداية 2 إلى 8 في نموذج الكود الجسم body للإجراء ، حيث يظهر كل منطقك . سيتم استخدام أي متغيرات تم تعريفها لوحدها هنا في الإجراء أيضاً ، كما بالمتغير theMessage على السطر 2 ، وينتهي الإجراء الفرعي دائماً بعبارة End Sub.

الدوال Functions

يختلف تركيب الدالة نوعاً ما عن الإجراءات الفرعية في دعمها لقيمة عائدة فقط .

```
01 Function IsPrime(ByVal source As Long) As Boolean
2:      ' تحديد فيما إذا المصدر عدد رئيسي.
3:      Dim testValue As Long
4:      If (source < 2) Then
5:          Return False
6:      ElseIf (source > 2) Then
7:          For testValue = 2 To source \ 2&
8:              If ((source Mod testValue) = 0) Then
9:                  Return False
10:             End If
11:         Next testValue
12:     End If
13:     Return True
14 End Function
```

كما مع الإجراءات الفرعية ، يظهر سطر التصريح للدالة أولاً (سطر 1) ، متبوعاً بالجسم (السطور من بداية 2 حتى 13) و بالاغلاق عبارة End Function(السطر 14) . يتضمن سطر التصريح تعريف نوع بيانات ما زائد بعد قائمة الوسيط . هذا هو نوع البيانات للقيمة النهائية كي تتم إعادتها إلى الكود المستدعي . استخدم قيمة العودة هذه في استدعاء الكود تماماً مثل أي قيمة أو متغير آخر . وكمثال على ذلك ، يستدعي السطر التالي الدالة IsPrime و يخزن نتيجته المنطقية Boolean في متغير ما :

```
primeResult = IsPrime(23)
```

و لتدل على عودة القيمة ، استخدم العبارة Return . يعمل نموذج الكود هذا على السطور 5 و 9 و13 . (قاعدة الفيجوال بيسك VB 6.0 القديمة التي تتيح لك أن تسند القيمة العائدة لاسم الدالة لا تزال تعمل.)

الخاصيات Properties

لقد ذكرت الحقول منذ وقت مبكر قليلاً ، والتي هي متغيرات أو ثوابت تظهر ضمن فئة ما ، ولكن خارج أي تعريف إجراء .

```
01 Class PercentRange
2:      Public Percent As Integer
03 End Class
```

إن الخاصيات مشابهة للحقول :حيث يتم استخدامها مثل متغير أو ثابت على مستوى الفئة class-level. ولكن يتم برمجتها مثل الدوال ، قابلة الوسيطات ، و مالكة قيم عائدة ، و متضمنة منطق كثير كما تطلب . و غالباً ما يتم استخدام الخاصيات لتحمي بيانات الفئة الخاصة مع المنطق الذي يتخلص من القيم الغير مناسبة . تعرف الفئة التالية خاصية وحيدة تدعم الوصل إلى الحقل المخفي(أو الخاص) ذو الصلة :

```
01 Class PercentRange
2:      ' يخزن نسبة مئوية من 0 إلى 100 فقط
3:      Private savedPercent As Integer
04 Public Property Percent( ) As Integer
05 Get
6:      Return savedPercent
07 End Get
08 Set(ByVal value As Integer)
9:      If (value < 0) Then
10:         savedPercent = 0
11:     ElseIf (value > 100) Then
12:         savedPercent = 100
13:     Else
14:         savedPercent = value
15:     End If
16 End Set
17 End Property
18 End Class
```

تحمي الخاصية Percent (السطور من 4 إلى 17) الوصول إلى الحقل savedPercent (السطر 3)، مصححة أي قيمة مزودة للمستدعي -caller supplied التي تجتاز المجال من 0 إلى 100 . تتضمن الخاصيات إسناد منفصل و استرداد المكونات ، و تستدعي محددات الوصول أيضاً . تعيد Get accessor (السطور من 5 إلى 7) قيمة مراقبة الخاصية إلى المستدعي . تدع Set accessor(السطور من 8 إلى 16) المستدعي يعدل قيمة الخاصية . تتضمن عبارة تصريح الخاصية (السطر 4) نوع بيانات والذي يلائم نوع البيانات المارة إلى داخل Set accessor (السطر 8) . هذا هو نوع البيانات لمجموعة القيم أو المستعادة من قبل المستدعي . لكي يستخدم هذا النموذج خاصية Percent ، أنشئ حالة ما من الفئة PercentRange، و بعدئذ استخدم الخاصية :

```
        Dim activePercent As New PercentRange
        activePercent.Percent = 107 ' An out-of-range Integer
        MsgBox(activePercent.Percent) ' Displays "100", not "107"
```

يمكنك أن تنشئ خاصيات للقراءة فقط read-only أو للكتابة فقط write-only تتضمن الكلمة المحجوزة القراءة فقط ReadOnly أو الكتابة فقط WriteOnly تماماً قبل الكلمة المحجوزة Property في عبارة التصريح (السطر 4) ، و ترك محددات الوصول accessor الغير محتاج إليها ، لا تحتاج الخاصيات لأن تربط بالحقول. يمكنك أن تستخدم الخاصيات لتجمع و ترتب أي نوع قيمة ، و تخزنه و تمثل عليه بأي طريقة تشاء .

أين تضع إجراءك Where to Put Your Procedures

بالعودة إلى الأيام القديمة الجيدة للفيجوال بيسك 6.0 ، فإنه بإمكان الإجراءات أن تظهر تماماً في أي مكان في ملفات كودك المصدري . عليك أن تفتح ملف مصدري ما ، اكتب دالي ما ، و اذهب : لقد كان ذلك سهلاً . مع الانتقال إلى الدوت نت ، فإنه جميع إجراءات الفيجوال بيسك يجب أن تظهر الآن ضمن فئة معرفة (أو تركيب أو وحدة برمجية).

```
Class Employee
        Sub StartVacation()
...
        End Sub
        Function TotalVacationTaken() As Double
...
        End Function
    End Class
```

عندما تنشئ حالات لفئتك فيما بعد في الكود ، يتم استدعاء الطرق مباشرة من خلال حالة الكائن .

```
Dim executive As New Employee
...
        executive.StartVacation()
```

ميزات أخرى للتحكم بالسياق Other Flow Control Features

تتيح لك الحلقات و العبارات الإضافية المتوفرة في الفيجوال بيسك إعادة توجيه كودك المبني على البيانات ، تتضمن اللغات عبارات أخرى قليلة والتي تدعك تضبط العمل في أكثر من طريقة مباشرة.

العبارة GoTo

تتيح لك العبارة GoTo القفز مباشرة إلى موقع آخر ما ضمن الإجراء الحالي . والغرض من القفز دائماً هو عنوان السطر line label،وهو موقع سطر مسمى في الإجراء الحالي . تظهر جميع عناوين السطور عند بداية السطر المنطقي ، و تنتهي بنقطتين .

```
PromptUser:
        GetValuesFromUser(numerator, denominator)
        If (denominator = 0) Then GoTo PromptUser
        quotient = numerator / denominator
```

في هذا النموذج ، تقفز العبارة GoToعائدة إلى العنوان PromptUserعندما يكتشف الكود قيمة بيانات غير مسموح بها . وتستمر معالجة البيانات مع السطر التالي مباشرة للعنوان PromptUser. لا يمكنك أن تستخدم نفس اسم العنوان مرتين في نفس الإجراء، على الرغم من أنك تستطيع أن تعيد استخدام أسماء العنوان في إجراءات مختلفة. إذا أردت ، ادخل عبارة منطقية أخرى على نفس السطر بوصفها عنوانك ،تماماً بعد النقطتين ، رغماً من أن كودك سيكون أسهل نوعاً ما حتى تقرأه إذا أبقيت العناوين على سطورها الخاصة .

```
LabelAlone:
        MsgBox("It's all alone.")
LabelAndCode: MsgBox("Together again.")
```

تستطيع كتابة عناوين متعددة في كودك كما تريد ، لكن العبارة GoTo هي واحدة من أولئك العناصر من الفيجوال بيسك التي تتم مراقبتها بشكل أقرب بواسطة وكالات البرمجة العالمية المزعجة ، مثل اللجنة العالمية للمحافظة على GoTo.تدقق تلك المجموعة أيضاً كتب الكومبيوترات باحثة عن مراجع ازدراء ليس لاسم تنظيماتها أن تجد أي شيء مثل ذلك في هذا الكتاب ، و لكن قضيته الجوهرية مفرطة في الاستخدام للعبارات التي يمكنها أن تقود إلى تغيير مسار الكود ، كالتالي :

```
Dim importantMessage As String = "Do"
        GoTo Step2
Step6:  importantMessage &= "AG!"
        GoTo Step7
Step3:  importantMessage &= "wit"
        GoTo Step4
Step2:  importantMessage &= "wn "
        GoTo Step3
```

```
Step5:   importantMessage &= "CK-G"
         GoTo Step6
Step4:   importantMessage &= "h I"
         GoTo Step5
Step7:   MsgBox(importantMessage)
```

يقول بعض الناس أن مثل هذا الكود صعب للقراءة . ويستدعيه آخرون لمهمة سرية . مهما تدعوه ، فإنه يصنع كود صعب جداً للمراجعة و للحفاظ عليه. من المحتمل أنه يجب عليك أن تحافظ على عين واحدة عند استخدامك لعبارات GoTo: وإذا لم تفعل ، فربما سيفعل ذلك شخص آخر .
تضع الفيجوال بيسك بحد ذاتها قيود على الاستخدام GoTo. ليس بإمكانك أن تقفز إلى أو من العبارات المتشابكة المحددة التي سوف تنتج في كود ابتدائي أو قيم بيانات بشكل غير مناسب. وعلى سبيل المثال ، لا يمكنك أن تقفز إلى المنتصف من العبارة For...Next من خارج العبارة ، بما أن متغير عداد الحلقة و مجالات البداية و النهاية لن تكون مبدئية كما ينبغي.

```
' هذه العبارة " إذهب إلى"ستفشل
        GoTo InsideTheLoop
        For counter = 1 To 10
InsideTheLoop:
            MsgBox("Loop number: " & counter)
        Next counter
```

على أية حال ، حالما تكون في داخل الحلقة ، فإنه بإمكانك القفز إلى عناوين السطر الذي يظهر أيضاً في الحلقة ، وهو مقبول أن تقفز خارج الحلقة مستخدماً GoTo. تفرض بعض التراكيب المتعددة السطور الأخرى قيود مشابهة .

عبارات العودة The Return Statement

ليس بإمكانك فقط أن تقفز في مكان قريب ضمن إجراء ما مستخدماً GoTo ، ولكن تستطيع أيضاً أن تقفز تماماً خارج الإجراء في أي وقت تريد مستخدماً العبارة Return، وعادة ، يغلق إجراء ما عندما تصل معالجة بيانات السطر الأخير للكود في الإجراء : تستمر بعدئذ معالجة البيانات مع الكود الذي استدعى الإجراء . تزود العبارة Return طريقة ما لتغلق الإجراء قبل الوصول إلى النهاية . في إجراء فرعي ، تظهر عبارة Return بحد ذاتها كعبارة ما مستقلة :

```
Return
```

في الدوال ، يجب أن تتضمن العبارة القيمة لتعاد استجابة لاستدعاء الكود : متغير ما ، محرف ما ، أو تعبير ما والذي يجب أن يصل نوع البيانات ذات القيمة العائدة المحددة للدالة .

```
Return 25
```

تحررت الدوت نت السابقة Pre-.NET من الفيجوال بيسك المستخدمة عبارة Exit لتترك بشكل مباشر إجراء ما . و لا تزال هذه مدعومة في الدوت نت. يوجد ثلاث تنوعات :

Exit Sub:
و هي تغلق إجراء جزئي ما.
Exit Function:
وهي تغلق دالة ما .
Exit Property:
و هي تغلق خاصية ما.

عند الخروج من دالة ما ، فإن العبارة Exit Function لا تتضمن طريقة ما لتخصص قيمة عودة . عليك أن ترتب قيمة العودة بشكل منفصل باسناد قيمة العودة للاسم من الدالة .

```
Function SafeDivide(ByVal numerator As Double, _
ByVal denominator As Double) As Double
        ' تجعل الإشارة # العدد من النوع البيانات المضاعف
        If (denominator = 0.0#) Then
            ' العودة بصفر في حال فشل القسمة
            SafeDivide = 0.0#
            Exit Function
        End If
        Return numerator / denominator
    End Function
```

عبارات التوقف و النهاية The End and Stop Statements

تقدم عبارات Stop و End توقف مباشر لتطبيق الفيجوال بيسك. تغلق العبارة End برنامجك مباشرة ، ملغية جميع الكود الإضافي و معالجة البيانات (على الرغم أنه تم توضيح مصادر مكتسبة محددة).
توقف العبارة Stop المعالجة فقط عندما تنفذ تطبيقك ضمن المصحح debugger ، كما في بيئة تطوير الفيجوال استديو . تحول Stop التحكم إلى البيئة ، متيحة للمطور أن يفحص ويغير البيانات والكود كل ما في الإمكان قبل الاستمرار مع البرنامج. إذا لم تتم مناقشة Stop في تطبيق مستقل منفَّذاً خارج المصحح ،إنه يحث المستخدم كي يصحح التطبيق مستخدماً أي مصحح منصب على شبكة الحاسوب . ليس من الضروري أن تقول ، إن المستخدم لن يتسلى .

الأحداث و معالجات الحدث Events and Event Handlers

إن الفيجوال بيسك هي لغة قيادة الحدث event-driven. وهذا صحيح بصورة خاصة لبرامج مكتوبة كي تنفذ على سطح مكتب ويندوز . بعد بعض الأوليات الهامة ، يتحكم المستخدم على العموم بجميع الأحداث في البرنامج . من يعرف ما سوف يعمله المستخدم المولع . ربما ينقر هنا . ربما تكتب هناك. من الممكن أن يكون فوضى وهرج . ولكن كل ما يفعله المستخدم ، سيعلّمك برنامجك حوله من خلال الأحداث events.
منذ الأيام الأولى للويندوز ، استخدِمت برامج سطح المكتب مضخة رسالةmessage pump لكي تتصل بالمستخدم و أحداث النظام لكودك . مدخلات الفأرة و لوحة المفاتيح ، و أحداث مولدة للنظام system-generated، و إشعارات أخرى من تدفق المصادر الخارجية إلى طوابير الرسالة message queue المشتركة للبرنامج . تطيل مضخة الرسالة هذه الرسائل واحدة تلو الأخرى ، وتفحصها ، تمدها بمساحات مناسبة من كودك.
في برمجة الويندوز التقليدية ، إنك تقوم بإظهار مضخة الرسالة بمهارة بنفسك ، مدخلاً الكود الذي يصنع استدعاءات مباشرة لإجرائية معالجة الحدث event-handling المبنية على نوع الرسالة . في برنامج الفيجوال بيسك (كل من الدوت نت و فيما مضى) ، تدعم اللغة مضخة الرسالة من أجلك . و تحلل الرسائل وكأنها مفرغة بواسطة المضخة خارج طابور الرسالة، و تقودها إلى الكود المناسب . ويظهر في الدوت نت الكود ضمن فئات . و حالما تملك فئة ما فرصة ما لتحلل الرسالة ، فإنه يمكنها أن تولد حدث ما ، والذي تتم معالجته أخيراً من قبل معالج الأحداثevent handler ، إن الإجراء الفرعي الذي كتبته كي يستجيب للفعل. وهذا الاستدعاء لمعالج يعرف بإطلاق*firing* حدث ما .
لذلك يوجد جزآن للحدث : (1) بعض الكود الذي يقرر إطلاق الحدث . (2) معالج الحدث الذي سوف يستجيب للحدث الذي تم إطلاقه.
إن الأحداث بالفعل هي استدعاءات غير مباشرة تماماً للإجراء . بدلاً من امتلاك الكود الرئيسي استدعاء إجراء فرعي آخر بشكل مباشر، فإنه يطلب من الدوت نت أن تستدعي الإجراء الفرعي الآخر من أجله ، ممررةً معاملات نسبية خاصة ربما يرغب الكود المستدعى تضمينها . لذلك ، لماذا تريد أن تعمل هذا بدلاً من صنع استدعاء مباشر لإجراء فرعي ؟ من أجل شيء ما ، تتيح لك هذه طريقة الغير مباشرة أن تضيف معالجات حدث لمدة طويلة بعد أن تتم كتابة كود إطلاق الحدث event-firing الابتدائي. هذا جيد ، بما أن كود إطلاق الحدث ممكن أن يكون في مجمع ثانوي تمت كتابته منذ سنين . المنفعة الثانية هي أن حدث واحد يمكنه أن يطلق معالجات حدث متعددة. عندما يتم إطلاق الحدث ، سيتم استدعاء كل معالج حدث ، وكل واحد يستطيع أن ينجز أي منطق خاص موجود في معالج الإجراء الفرعي.
يمرر الكود الذي يطلق الحدث بيانات خاصة event-specific بالحدث إلى معالج أو معالجات الحدث الهدف من خلال قائمة وسيطات الحدث . من أجل أن يعمل استدعاء الإجراء الفرعي الغير مباشر ، يحتاج معالج الحدث لأن يحوي العدد الصحيح من المعاملات النسبية، في الترتيب الصحيح ، كل نوع من البيانات المحددة و المتوقعة . تعرف عبارة الحدث Eventهذا العقد (الاتفاق) بين الحدث و المعالج .

```
Public Event SalaryChanged(ByVal NewSalary As Decimal)
```

تعرف عبارة الحدث Eventحدث ما مسمى ب SalaryChanged مع معامل نسبي وحيد ، ذو قيمة عشرية Decimal.إن رغبة أي معالج حدث في أن يراقب الحدث يجب أن يلائم توقيع المعامل النسبي .

```
Sub EmployeePayChanged(ByVal updatedSalary As Decimal)...
```

يمكن أن تحدث الأحداث Events من أجل أي سبب تعتبره ضرورياً :حيث لا تحتاج الأحداث أن تقيد إلى مستخدم أو أعمال نظام . في نموذج هذه الفئة، يتم إطلاق حدث ما كل مرة يتم عمل تغيير لراتب الموظف . تنجز العبارة RaiseEvent الإطلاق الفعلي للحدث ، مخصصة الاسم للحدث ليتم إطلاقه ، و مجموعة من المعاملات النسبية في الأقواس .

```
Public Class Employee
        Public Name As String
        Private currentSalary As Decimal
    Public Property Salary() As Decimal
        Get
            Return currentSalary
        End Get
        Set(ByVal value As Decimal)
            currentSalary = value
            RaiseEvent SalaryChanged(currentSalary)
        End Set
    End Property
    Public Event SalaryChanged(ByVal NewSalary As Decimal)
End Class
```

لا تتم إضافة معالجات الحدث بشكل مباشر إلى الفئة . عوضاً عن ذلك ، تتم إضافتها إلى حالة ما للفئة . الحالة ، مصرح عنها كحقل فئة ما ، يجب أن يتم تعريفها استخدام الكلمة المحجوزة WithEvents الخاصة ، التي تخبر الفيجوال بيسك أن هذه حالة سوف تعالج الأحداث .

```
Public WithEvents MonitoredEmployee As Employee
```

معالجات الحدث هي إجراءات فرعية مألوفة ، ولكنها تتضمن الكلمة المحجوزة Handles لتشير أي حدث تتم معالجته .

```
    Private Sub EmployeePayChanged(ByVal updatedSalary As Decimal)Handles
MonitoredEmployee.SalaryChanged
        MsgBox("The new salary for " & MonitoredEmployee.Name & " is " & updatedSalary)
    End Sub
```

كل ما يحتاجه هو شيء ما لاستئناف الحدث .

```
    Public Sub HireFred()
        MonitoredEmployee = New Employee
        MonitoredEmployee.Name = "Fred"
        MonitoredEmployee.Salary = 50000 ' Triggers event
    End Sub
```

عندما يكون الراتب مجموعة ، تطلق خاصية Salary للفئة Employee الحدث SalaryChanged مستخدمة أمر الفيجوال بيسك RaiseEvent. و هذا يولد استدعاء ما لمعالج الحدث EmployeePayChanged، الذي يعرض بالنهاية الرسالة . الأحداث المبنية داخل الفئات Windows Forms في الدوت نت NET.تعمل تماماً مثل هذا ، ولكن بدلاً من أن تراقب معي زيادة الراتب . فإنها تترقب نقرات الفأرة و ضربات لوحة المفاتيح . تستخدم جميع أحداث النظام هذه توقيع معاملات نسبية ما مشترك.

```
Event EventName(ByVal sender As System.Object,ByVal e As System.EventArgs)
```

يطابق المعامل النسبي المرسل sender الحالة للكائن الذي يطلق الحدث ، في حال احتاج المستدعي اختبار أعضاءه. إن المعامل النسبي e هو كائن ما يتيح للمستدعي أن يرسل بيانات خاصة بالحدث event-specific إلى المعالج من خلال حالة فئة وحيدة . لا تملك الفئة System.EventArgs الكثير على طول مع الأعضاء ، و لكن العديد من الأحداث تستخدم فئة بديلة و التي تشتق من System.EventArgs.

فضاءات الأسماء Namespaces

الفئات و التراكيب و الوحدات البرمجية و العدادات و الواجهات و التفويضات ـ أنواع الدوت نت الرئيسية ـ لا تعوم تماماً فقط في كود تطبيقك .و عليها جميعاً أن تصنف و تدار في فضاءات أسماء namespaces، يزود فضاء الأسماء هرم ما لأنواعك ، جزء من شقة مملوكة في بناية شجرية الشكل tree-shaped حيث يملك كل نوع بيت ، بعض هذه البيوت (أو العقد) ، مثل النظام System ، تحصل على مجموعة كبيرة مناسبة مع جميع أنواع هذه العائلات الساكنة هناك . الآخرون ، مثل System.Timers ، ربما تملك بضعة أنواع تسكن في مساكنها الكافية . ولكن كل نوع يجب أن يعيش في الهرم.

في جذر الهرم ذاته يوجد Global ، و ليس عقدة بذاته ، و لكن كلمة فيجوال بيسك محجوزة والتي تشير إلى جذر كل الجذور . بإمكانك أن تدخل Globalعند إرجاع فضاءات الأسماء الخاصة بك ، ولكن استعمالها مطلوب فقط ومتى تتركها فسوف تسبب فوضى بين فرعي فضاء أسماء.
بشكل مباشر تحت Globalيوجد فضاءات أسماء عالية المستوى top-level قليلة ، متضمنة System وMicrosoft. يحوي كل فضاء أسماء عالي المستوى top-level فضاء أسماء ثانوي ، و كل من تلك تحوي فضاءات أسماء مستوى ثالث third-level إضافية ، و هكذا . يتم إرجاع عقد فضاء الأسماء بنسبة أحدها إلى الآخر مستخدماً ترميز النقطة"dot" .

```
System.Windows.Forms
```

هذا يحدد فضاء أسماء Forms على المستوى الثالث . بإمكانك أيضاً أن تكتب :

```
Global.System.Windows.Forms
```

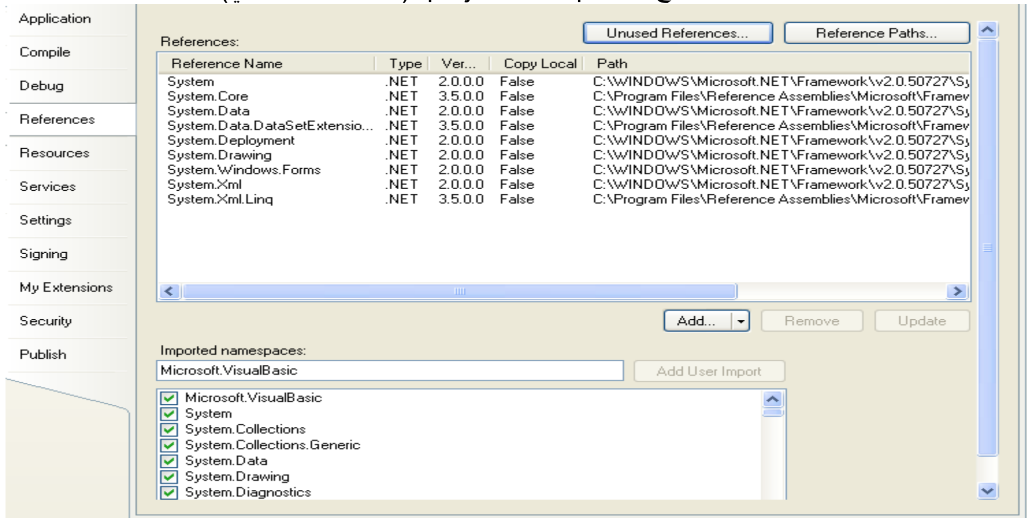
الذي يعني نفس الشيء . أيضاً يتم دعم فضاء أسماء نسبي:

```
Forms
```

على أية حال لاستخدام فضاءات أسماء نسبي ، عليك أن تخبر كود الفيجوال بيسك لكي يتوقعها . يوجد فضاءات أسماء عديدة هناك ، و ربما يوجد أماكن فضاءات أسماء Forms عديدة في الهرم.

إسناد مؤشر إلى فضاءات الأسماء Referencing Namespaces

قبل أن يتم استخدام فضاءات الأسماء في كودك ، كان من الواجب إسناد دليل لها أو مرجع referenced وجلبها imported بشكل اختياري . يعين إسناد دليل أو مرجع إلى فضاء الأسماء ملف المجمع DLL الذي يحوي أنواع فضاءات الأسماء تلك. اعمل كل من هذين الفعلين من خلال التبويب"مراجع References" من نافذة خاصيات المشروعproject's Properties. (انظر الشكل التالي)



في الحقيقة ، إنك لم تشير إلى فضاءات الأسماء في DLL ، ولكن بالأحرى الأنواع ، و جميعها يحدث لأن تقطن في فضاء أسماء خاص . على أية حال ، للنوع DLLs الرئيسي المزود بإطار عمل الدوت نت ، إنه ضمنياً نفس الشيء . في الحقيقة ، حتى الميكروسوفت سمت العديد من ال DLLs لمجارات فضاءات الأسماء التي تتضمنها. يتضمن System.dll أنواع ضمن فضاء الأسماء System. تتضمن System.Windows.Forms.dll أنواع معينة لتطبيقات Windows Forms ، و تظهر كل هذه الأنواع في فضاء الأسماء System.Windows.Forms أو أحد فروعها .
إذا لم ترجع DLL في مشروعك ، فإنه ولا شيء من أنواعها سيكون متوفراً لديك في كودك . تحمّل الفيجوال استديو مراجع عديدة داخل مشروعك بشكل آلي مبنية على نوع المشروع الذي تنشئه أنت . و يظهر الشكل السابق المراجع الافتراضية التسعة المحتواة ضمن تطبيق Windows Forms وهي :
System، System.Core، System.Data، System.Data.DataSetExtensions، System.Deployment، System.Drawin، System.Windows.Forms، System.Xml، وSystem.Xml.Linq .

حالما تعمل على إرجاع مكتبة فئات ما (أو أنواع أخرى) في كودك ، فإنك تتمكن من الوصول لأي من فئاتها بكتابة الاسم الكامل لفضاء أسماء تلك الفئة . على سبيل المثال ، الفئة للفورمالتي تكون على الشاشة on-screen يتم إرجاعها بواسطة System.Windows.Forms.Form. وتلك ثلاث مستويات إلى أسفل الهرم ، وبعض الفئات هي أعمق حتى .
لتتجنب كتابة جميع فضاءات الأسماء الطويلة هذه عدة مرات متكررة ،فإنه تتضمن الفيجوال بيسك ميزة imports. حالما يتم جلب فضاء الأسماء ، فبإمكانك أن تصل لأي من الأنواع في فضاء الأسماء بدون تحديد اسم فضاء الاسم . إذا جلبت فضاء الأسماء System.Windows.Forms ، عليك فقط أن تكتب "Form"لتدخل الفئة Form. يظهر النصف السفلي من الشكل السابق كيفية ترتيب هذه الواردات من خلال خصائص المشروع .إن القائمة "Imported namespaces" تظهر جميع فضاءات الأسماء المرجعية المتوفرة . ببساطة ضع علامة صح على الفضاءات التي ترغب أن تجلبها لقد تم وضع علامة صح على System.Windows.Forms مسبقاً وبشكل افتراضي في تطبيقات Windows Forms.
بإمكانك أيضاً أن تجلب فضاء أسماء مباشرة في كودك المصدري . استخدم العبارة Imports من بداية ملف الكود المصدري :

```
Imports System.Windows.Forms
```

تدعم العبارة Imports اختصارات فضاء أسماء ، أسماء قصيرة و التي تمثل فضاء الاسم الكامل في كودك . باستخدام العبارة :

```
Imports Fred = System.Windows.Forms
```

تتيح لك إرجاع الفئة Form مثل"Fred.Form." و ليس مثل قائمة الإيرادات في خصائص المشروع ، التي تؤثر على كامل المشروع ، تؤثر العبارة Imports في ملف كود مصدري وحيد والمكتوبة فيه فقط .

فضاءات الأسماء في مشروعك Namespaces in Your Project

بشكل افتراضي ، تظهر جميع الأنواع و الفئات في مشروعك في مستوى فضاء أسماء أعلى top-level والذي يواجه اسم مشروعك . من أجل تطبيقات Windows Forms ، و يدعى فضاء الأسماء الافتراضي هذا ب WindowsApplication1. ولتحدد مستوى فضاء أسماء أعلى مختلف top-level ، غيّره من خلال المؤشرApplication لخصائص المشروع ، في الحقل "Root namespace" . و تظهر جميع الأنواع في مشروعك في فضاء الأسماء هذا : إذا حددت فضاء أسماء مزود من قبل ميكروسوفت Microsoft-supplied موجود كفضاء أسماء جذري لمشروعك ، سوف تظهر جميع فضاءات الأسماء في مزيج فضاء أسماء مخصص في الأنواع الموجودة مسبقاً . من أجل التطبيقات المستقلة ، هذا المزيج سوف يكون واضحاً فقط من كودك .
من فضاء الأسماء الجذر ، يمكنك أن تضع أنواع ضمن فضاءات أسماء ثانوية باستخدام العبارة Namespace.و Namespaceهي عبارة مقطع ما تنتهي بالعبارة End Namespace. سوف يتم احتواء أي أنواع أنشأتها بين العبارات Namespaceو End Namespace في فضاءات الأسماء الثانوية تلك. و مثالاً على ذلك ، إذا كان فضاءك الأسماء الجذر هو WindowsApplication1، فإن العبارات التالية تنشئ فئة ما و التي اسمها الكامل هو WindowsApplication1.WorkArea.BasicStuff.BusyData :

```
Namespace WorkArea.BasicStuff
        Class BusyData
...
        End Class
    End Namespace
```

بإمكانك أن تدخل كعبارات فضاء أسماء متعددة في كودك كما تدعي الحاجة . ويتم تزويد ذلك بإدخال فضاءات أسماء أيضاً :

```
Namespace WorkArea
        Namespace BasicStuff
            Class BusyData
...
            End Class
        End Namespace
    End Namespace
```

فضاء الأسماء الخاص بي The My Namespace

قدمت الفيجوال بيسك 2005 فضاء أسماء جديد"My" في مستوى أعلى top-level ، مصمماً لتبسيط مهام برمجية مشتركة . وأضافته الميكروسوفت إلى اللغة جزئياً لترسم بسالة عرض فيجوال بيسك الإصدار 6.0 داخل ملف الدوت نت . و لكن معظمه ليس حقيقياً حيث أنه تمثيلي . يجمع My عموماً ميزات مستخدمة والتي تنشر حالياً حول مكتبة فئة إطار العمل Framework Class Library (FCL) ، و تضعها في هرم صغير mini-hierarchy للدخول المناسب . إنها بالفعل ليست أكثر تعقيداً من ذلك . و يتم تنظيم الهرم بإحكام ، مع أقسام المستخدم ، التطبيق ، و معلومات الكمبيوتر الخاصة . يتم استخدامه تماماً كأي جزء آخر من إطار العمل ، رغماً من أنك لا تستخدم الكلمة المحجوزة Imports لتدخل عناصرها بطريق نسبية .
على أية حال ، إن My سهلة جداً للاستخدام ، لتظهر عدد الإصدار لتطبيقك ، على سبيل المثال ، استخدم العبارة التالية :

```
MsgBox(My.Application.Info.Version.ToString)
```

تكون بعض مناطق فضاء الأسماءMy ديناميكية : و تضاف فئات أو تنقل عندما تعدّل كودك المصدري . في تطبيقات Windows Forms ، يتضمن الفرع My.Forms مداخل لكل صيغة من صيغ المشروع . عندما تضيف صيغاً جديدة ، عندها تتم إضافة مدخلات جديدة بشكل أوتوماتيكي . ثم يصنع المشروع My.Forms عنوان المرجع لكل صيغة متاحة للاستخدام في كودك .

```
My.Forms.Form1.Text = "Welcome"
```

مشروع المكتبة:The Library Project

يعمل هذا المشروع على إدارة قاعدة بيانات كتب وبنود وسائط أخرى،ويتحكم بكيفية نقل هذه البنود بين رفوف الكتب والزبائن.يحتاج البرنامج أن يمتلك ميزات خاصة بالزبائن والإدارة.وسيتضمن تقارير متنوعة ،بما فيها طباعة فاتورة بنود المخرجات للزبون.وأهم من هذا كله،يحتاج أكواد الطباعة وقراءة تعريف الأدوات(الوسائط المركبة على الكمبيوتر)،إنه يبدو كثير على أن يقوم بعمله شخص واحد فقط،وإنه مشروع قابل للزيادة في الحجم.ولكن لن أقوم بعمله بنفسي،سوف تساعدني.فمع بعضنا البعض،ومن خلال صفحات هذا الكتاب،سوف نصمم كلانا هذا البرنامج،ونطور كوده،وسنجلب المتعة للمستخدمين.ما تبقى من مستندات هذا الفصل هي الميزات الرئيسية لتطبيق إدارة المكتبة.

ميزات بند المكتبة:Library Item Features

نظام المكتبة سيعمل على إدارة بيانات مفصلة بالكتب وبنود الوسائط الأخرى،إيجاد هذه البيانات،وإدارة وترتيب التفاصيل والحالات بالنسبة لكل نسخة من أي بند.ولجعل هذا حقيقة،فإن برنامج المكتبة سيعمل التالي:

- ✓ يسمح للمدراء أو الزبائن من البحث عن بنود حالية في البيانات المفصلة.سيسمح البرنامج بالبحث بالاعتماد على العديد من الخاصيات المختلفة لكل بند.
- ✓ دعم طرق البحث المتنوعة ،من ضمنها بواسطة العنوان،بواسطة اسم المؤلف،بواسطة الموضوع أو تصنيف العنوان(مثل كتب علوم الإنسان...)،أو بواسطة كلمات مفتاحية متنوعة،أو بواسطة اسم دور النشر التي لها علاقة،أو بواسطة اسم الرقم المسلسل أو المجموعة التي تحتوي ذلك البند،أو بواسطة رقم كود التعريف المرتبط بالبند الحقيقي.
- ✓ تحديد نتائج البحث بواسطة موقع البند،أو بوساطة نوع الوسائط(كتاب،سيدي،ديفيدي،إلخ).
- ✓ دعم التعريف واستخدام المواقع المادية المميزة.فالزبون لديه كتب ووسائط تخزين في ثلاث مواقع مختلفة ضمن البناء،ومن ضمنها خزانة تخزين صغيرة للبنود النادرة الاستخدام.
- ✓ عرض تفاصيل البند المستخرج ضمن واجهة نمطية استعراضية معروفة(شائعة).على سبيل المثال،عند البحث عن كتاب بواسطة العنوان،ينقر المستخدم على اسم المؤلف للوصول إلى جميع البنود الأخرى لذلك المؤلف.
- ✓ السماح بالوصول إلى كل بند مكتبي من خلال مسح كود تعريف ما.كما هو شائع مع معظم مكتبات هذه الأيام،فالبنود في مجمع هذه المكتبة لديها ملصقات أكواد تعريفية.والتي تخدم كمعرف وحيد ومفرد لكل نسخة لبند مستقل.

ميزات الزبون Patron Features

بالإضافة إلى الكتب والبنود الأخرى،يدير البرنامج قائمة من الزبائن،"زبائن"المكتبة والذين يتم الترخيص لهم بتفحص(أو مراجعة) بنود المخرجات.لدعم التفاعل مع الزبائن،سيتضمن التطبيق هذه الميزات الخاصة بالزبون:

- يمكن للزبائن من مراجعة البنود دون مساعدة المدراء وأمناء المكتبة.،فبإمكانهم استخدام ماسح كود التعريف لمسح بنود المكتبة وكرت مكتبة الزبون وبنود المكتبة.
- يمكن أن يتم تفحص بنود الزبائن،ومراجعتها في فاتورة المكتبة.
- يتم إسناد رقم خاص" PIN "لكل زبون والذي يتصرف ككلمة مرور.
- تحدد"أنواع الوسائط"لبند ما فترة (عائداته)أو مخرجاته checkout (ولاحقاً عملية إعادة تجديده أو تحديثهrenewal)
- يمكن للعملاء من عرض سجل المكتبة الخاص بهم،ومن ضمنه كل الكتب المخرجة حالياً،وقائمة الغرامات المدينين بها للمكتبة.
- إذا رخص على بند معين،فبإمكان الزبون تحديث بند ما قد قام بمراجعته حديثاً.
- المساعدة المركزية على الشبكة للزبون تكون متاحة من خلال المفتاح F1القياسي .وملف المساعدة يتضمن معلومات غير الميزات الإدارية،مما يقلل من التجريبexperimentation .
- يمكن تقسيم الزبائن ضمن "مجموعات الزبائن"من أجل التقارير والمعالجات المريحة(المناسبة) لطاقم الإدارة.administrative staff .

الميزات الإدارية:Administrative Features

من ضمن المدراء أمناء المكتبات،الهيئة الإدارية،وأشخاص آخرين بحاجة لإمكانية وصول متقدمة لميزات التطبيق.وهم المستخدمين الرئيسين للنظام،وليس العملاء.يتضمن التطبيق ميزات خاصة بالمدراء التالية:

- o توفر ميزة "تسجيل الدخولlogin " الوصول إلى الميزات الإدارية للتطبيق.يمكن فقط للمستخدمين المفوضين من تسجيل الدخول من خلال كلمة المرور المخصصة.وميزة تسجيل الدخول عادة تكون مخفية عن المشاهدة بالنسبة للعميل العادي.
- o يمكن للمدراء من مشاهدة تفاصيل العملاء تماماً كما يشاهدها العميل،ولكن لديهم أيضاً إمكانية الوصول إلى تفاصيل العملاء الإضافية.وعلى وجه التخصيص بإمكان المدراء إضافة عملاء جدد وإدارة ذاتيتهم وتفاصيل دراسية أو إحصائية.وبإمكان المدراء أيضاً تعطيل تسجيل عميل ما لمنع مخرجات بنود أكثر.
- o يجمع ويدير المدراء الغرامات المالية fines الخاصة بالعملاء،ومن ضمنها إمكانية إضافة غرامات غير قياسية أو صرف الغرامات غير المدفوعة unpaid fines .
- o يحدد المدراء سجلات كل بند والمدارة بواسطة قاعدة بيانات فاتورة النظام.وهذا يتضمن قواعد كل بند،مثل العنوان والمؤلفين.فكل بند يتضمن نسخة واحدة أو أكثر،والتي تمثل البنود المادية والتي يمكن مراجعتها.ويتم إسناد أكواد التعريف للنسخ.
- o إلى جانب البنود والنسخ،يحدد المدراء جميع القوائم والقيم المدعومة،ومن ضمنها أسماء المؤلفين والتصنيفات،قائمة أنواع الوسائط،الناشرين، أسماء الكتب المتسلسلة،أكواد الحالة والتي تحدد ترتيب نسخة كل بند،والموقع.
- o المدراء المختارين(الخصوصيين) بإمكانهم إضافة،تحديث،أو إزالة حسابات مدراء آخرين.يتضمن كل حساب ميزة إعدادات تفويض خاصة (مجموعة الحقوق).
- o بالإضافة إلى مسح أكواد التعريف،يمكن للبرنامج من مساعدة المدراء في تصميم وطباعة أكواد تعريف البنود والعملاء.
- o تسمح عملية إدارة البرنامج البسيطة لهيئة الإدارة من معالجة البنود المتأخرة الدفع والغرامات بالاستناد على قواعد نظامية.

- يسمح التطبيق من أن يتم إضافة العطل وحفظها.فعندما يراجع عميل ما كتاب ،يضبط البرنامج تاريخ المستحقات للبند لتجنب العطل.
- يوفر مركز التعليمات الخاص بالإدارة مساعدة لتطوير ميزات التطبيق من خلال نفس المفتاح المتاح للعملاء.
- يتضمن التطبيق بعض التقارير الإدارية الأساسية،والمقدرة على ربط "plug in"التقارير عند الحاجة في المستقبل دون الحاجة لتحديث البرنامج نفسه.

التطبيق ككل:The Application As a Whole

إلى جانب الميزات الأساسية للبرنامج التي تم اختبارها بواسطة العملاء والمدراء يوجد العديد من المتطلبات الأخرى:

- البرنامج سهل الاستخدام بالنسبة للمستخدم وسهل الاستعراض،وخاصة للعملاء،دون الكثير من التجريب والتدريب trainingأو المساعدة.
- يخزن التطبيق بياناته في قاعدة بيانات مخدم سكول SQL Server database.
- نشر التطبيق يتم عمله بواسطة هيئة الإدارة والتي يكون لديها امتيازات إدارية محلية،لذلك فإن حزمة تنصيب ويندوز تكون كافية.
- يستخدم تركيب(إعداد) التطبيق طرق XMLالقياسية .

ماعدا هذه المتطلبات العامة والميزات الخاصة،فقد عملت على منح تصميم حر،ولكن من أين ستأتي قائمة المتطلبات؟إنها ستأتي من المستخدمين،أصحاب التطبيق.فهو يلبي احتياجاتهم_احتياجات زبائني،الذين سيستخدمون التطبيق يوم بعد يوم- وهذا ما يحدد قائمة من المتطلبات.

احتياجات المستخدمين The Needs of the Users

بالعودة إلى أيام الكمبيوتر الأولى،فلم يكن هناك مستخدمين،فمن يحتاج لمستخدمين؟للمستخدمين الكثير من الحاجيات،ومعظمها لايمكن سدها بواسطة الكمبيوتر.ولكن بالنسبة للتي يمكن عملها،تأتي الحاجيات في خمسة أقسام:البيانات والمعلومات،العمليات،إمكانية الاستخدام، العموميات،والحاجيات الخاصة بالمشروع.تتضمن عملية التصميم اختبار هذه الحاجيات والمراسلات الناتجة عن هذه الحاجيات ضمن البرنامج المنتج.بتفحص البيانات الحالية والإجراءات، إجراء مقابلات مع المستخدم،وإتمام طرق استخلاص الحاجات الأخرى،فأنت تجمع التفاصيل التي تحتاجها لنحت الحل المناسب.

البيانات والمعلومات Data and Information

إن مقدرتك على توفير إمكانية وصول خاصة ومناسبة للبيانات والمعلومات المطلوبة من قبل المستخدم هو مايجعلك مبرمج محبوب جداً.معظم المستخدمين يأخذون وقت طويل لكي ينسجموا تماما مع الكمبيوتر.فهم يحفظون معلوماتهم على 3-5 كروت تصنيف،أو لبادة أو لفافة ورقية أو مخطوطات ورقية أو ما شابه،وبالتالي فلديهم سبب ما للانتقال إلى الكمبيوتر كوسيلة تخزين: وهي أنه وسيلة مريحة.البيانات هي المعلومات الصرفة(المجردة)والمخزنة وبواسطة برنامجك:أسماء،أعداد،صور،أو أي قيمة مستقلة قائمة بذاتها.أما المعلومات فهي البيانات في سياق:سجل الزبون،أو في ترتيب معين، مظهر المنزلقات(طبقات معروضة على سطح ما).عندما توفر برنامج نوعي يرتقي بالبيانات إلى مستوى المعلومات،فأنت توفر مستوى مريح ومناسب لاحتياجات المستخدمين .

المعالجةProcess

عندما يطلب المستخدم استرجاع بياناته من الكمبيوتر،فلديك ثلاث خيارات:

- إفراغ كل بايت مفرد من البيانات إلى الشاشة،الطابعة،أو الديسك،وتدع المستخدم يصنفها،عملياً،هذا هو النظام الذي لدى بعض المستخدمين قبل أن يبدؤوا باستخدام الكمبيوتر.
- حماية البيانات من يتمكن المستخدمين من الوصول إليها،بالتشديد أو الإصرار على تزويد كلمة مرور صحيحة أو تاريخ الانتهاء،أو أن تكون البيانات غير متاحة ولايمكن الوصول لها .من خلال "إلغاء الأمر،إعادة المحاولة،فشل".عملياً،هذا هو النظام الذي لدى بعض المستخدمين الآخرين قبل البدء باستخدام الكمبيوتر.
- جلب البيانات كمعلومات،على هيئة قابلية الاستخدام والوصول.

على الرغم من أن كلا الخيارين الأول والثاني هما عملياً تجريبيان،فالخيار الثالث هو الأفضل.وكمية البيانات الممنوحة والتي من المحتمل أن يعمل تطبيقك على إدارتها،فسيكون عليك توزيع التفاعل معها بت في كل مرة،وبتتابع مناسب.وهذه هي المعالجة *process*.

من خلال تنفيذ معالجة محققة،فأنت تتحكم ليس فقط ببيانات المستخدم،ولكن أيضاً التفاعل المنهجي مع البيانات.معظم المستخدمين بحاجة إلى مزود أو استخلاص فقط جزء صغير من البيانات كل مرة.ولكن عندما يعملون ذلك،فسيكون هذا ضمن سياق بعض المعالجات.على سبيل المثال،فلشغل موقع أو مكان ،فالمستخدم:(1)يعمل على إدخال وتأكيد معلومات اتصال الزبون،(2)إدخال أو تحديث تفاصيل الطلب،و(3)يطبع أو يتصل إلكترونياً بمعلومات الطلب لكي يصبح تام.فتطبيقك يدير هذه المعالجة للخطوات الثلاث.

قابلية الاستخدام Usability

إذا كان برنامجك يحضر البيانات والمعلومات للمستخدم وبترتيب معين أو بطلب معين ،ولكنه صعب الاستخدام فسيكرهك المستخدمين.وسوف يشمئزون loathe منك.وسينسجون قصص حولك،سواء كانت حقيقية أم لا.

كمبرمج ،فإن عملك أن تجعل الكمبيوتر والبرمجيات التي تعمل عليه،قابلة للاستخدام قدر الإمكان.وعلى الرغم من عدم مقدرتك على التحكم بالعديد من الميزات الأساسية للنظام،فتبقى أنت الملك عندما يتعلق الأمر بالبرمجيات الخاصة بك.كلما كانت السهولة وقابلية الاستخدام في تصميم برنامجك،كلما كان المستخدمين أكثر سعادةً. ولكن يجب عليّ أن أحذرك،فكلما كانت السهولة أكبر للمستخدم كلما كان هذا يعني العمل أكثر على التطوير (البرمجة).دائماً، أقدم لك النصائح التالية فيما يخص أي برنامج:

- لا تضع الكثير من المعلومات في نموذج واحد.وعندما ترتاب،أنقل بعض المعلومات إلى نموذج أخر.
- قم بجلب المعلومات والبيانات الضرورية فقط للمستخدم بشكل افتراضي،واعرض المعلومات الإضافية فقط إذا طلبها المستخدم.
- أجعله سهل من أجل المستخدم لكي يصل ويحدث البيانات،ولكن اسمح للبرنامج اسمح للبرنامج من العمل بشكل مناسب دون الحاجة لهذه البيانات.
- استخدم النصوص ،الصور،والألوان فيما ينفع المستخدم.
- بسط التطبيق بحيث تبدو مستندات المستخدم(مستندات المساعدة أو المعلومات)غير ضرورية.
- دائماً وفر مستندات المستخدم.وأجعلها بسيطة بشكل كافي بحيث يصبح الاتصال من أجل الدعم الفني غير ضروري.

هذه القواعد عامة وكافية للعمل مع أي نوع من التطبيقات،وهي من وجهة نظرك كافية بالنسبة لنا كمبرمجين وهامة لهم كمستخدمين.

العموميات (الخصائص المشتركة)Commonality

تتطلع touts ميكروسوفت باستمرار لتجديد(الابتكار*innovation*)،والمقدرة على الابتكار قد نقلت المنتجات البرمجية للأمام بخطوة هائلة.ولكن لسوء الحظ،على المستخدمين إمكانية معاملة فقط الكثير من الابتكار كل مرة.بالنسبة للبرمجيات:فحتى مع البرامج المبتكرة والجديدة يجب المحافظة على بعض العموميات مع نظام التشغيل،وبما يتوافق مع البرامج الأخرى التي يتم تثبيتها على الجهاز،وكما تتحدث مع المستخدمين حول احتياجاتهم وتفكر بأكبر تقدم في تقنية البرمجيات والتي ستعمل على توفيرها،فلا تنسى العموميات.كما أنك لاتنسى المتطلبات الجوهرية للمستخدمين:فلا تغمر المستخدمين بالطرق الجديدة لعمل المهام التي يفكرون بأنه يتم عملها بطرق تقليدية دائماً يقومون بها بنفس الأسلوب،فما يحتاجه المستخدمون هو الاستقرار consistency في طريقة الأداء.

المشروعProject

يحتوي دليل هذا الفصل الملفات التالية:

مستند.اتفاقية المشروع Project Agreement.doc

هذا هو مستند المشروع الرئيسي والذي يحدد ميزات المشروع الكامل.وتحظى الاتفاقية بتمثيل كل من المطور والمستخدم.والخروج عن هذه الوثيقة يحدث من خلال معالجة"طلب التغيرChange Order " فقط.

مستند.طلب التغير Change Order.doc

يتم استخدام هذا المستند لتعديل المشروع الأصلي من خلال معالجة "طلب التغير".عند استخدام هذه الوثيقة،فهي تتضمن وصف للتغير الذي يجب عمله على المشروع،وأي جدولة وتكاليف سوف تتأثر.

مستند.قبول المشروعProject Acceptance.doc

يتم استخدام هذه الوثيقة عندما يكتمل المشروع.والمستخدم جاهز لقبول المنتج النهائي.وهذا المستند يضم عناصر"اختبار معيار القبول Acceptance Criteria Testing "و"قبول المشروع" المشروحة مسبقاً في هذا الفصل.

ولك الحرية التامة في استخدام هذه الوثائق لدعم مشروعك الخاص.

تصميم قاعدة البيانات Designing the Database

إذا كان لديك بيانات فتحتاج لوضعها في مكان ما.وهل هناك مكان أفضل من وضعها في قاعدة بيانات؟قاعدة البيانات هي تجمع من البيانات المنظمة لسهولة استخراجها بواسطة الكمبيوتر.

قواعد البيانات العلائقية: Relational Databases

تعمل قواعد البيانات العلائقية على تجميع البيانات في جداول *tables*،وكل منها يخزن مجموعة خاصة من السجلات *records* غير المرتبة.من أجل الراحة،يتم إحضار الجداول كشبكة من قيم البيانات،حيث أن كل صف *row* يمثل سجل وحيد وكل عمود *column* يمثل حقل *field* ثابت والذي يظهر في كل سجل.يمثل الجدول التالي جدول طلبيات،مع سجلات منفصلة السطور فكل سطر يمثل بند طلبية.

Customer

Record ID	Order ID	Customer ID	Name	Product ID	Product	Price	Quantity
92231	10001	AA1	Al Albertson	BEV01COF	Coffee	3.99	3
92232	10001	AA1	Al Albertson	BRD05RYE	Rye bread	2.68	1
92233	10002	BW3	Bill Williams	BEV01COF	Coffee	3.99	1
92234	10003	BW3	Bill Williams	BEV01COF	Tea	3.99	2
92235	10004	CC1	Chuck Charles	CHP34PTO	Potato chips	0.99	7

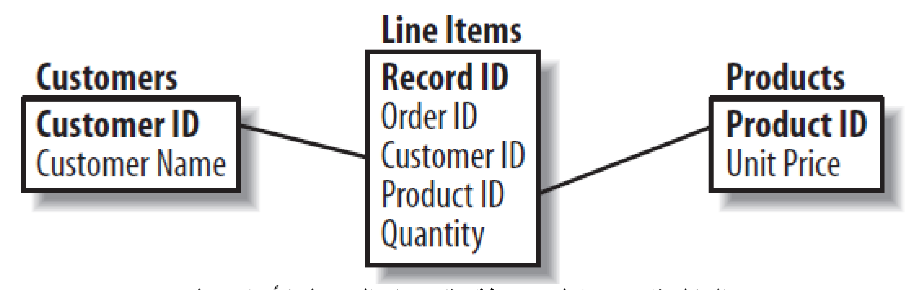
إن وضع جميع معلوماتك في جدول مناسب تماماً.فالبيانات الهامة تظهر من خلال نظرة سريعة في ترتيب مناسب وجميل،ومن السهل ترتيب النتائج بالاعتماد على عمود معين.لسوء الحظ،هذا الجدول من الطلبيات لديه الكثير من أسماء الزبائن Customer ID المكررة وأسماء المنتجات Product تتكرر أيضاً عدة مرات. وبالإضافة إلى معرف اسم المنتج product ID، فيشير BEV01COF إلى قهوة،وواحد من أسطر القائمة يشير إلى شاي،والعديد من المشاكل تكون ملازمة للبيانات الموضوعة في ملف *flat file* وحيد لجدول قاعدة بيانات database table. وبالتالي كان الحل فيما يسمى بالتطبيع " *normalization*." وذلك بتقسيم البيانات ضمن جداول منفصلة مع مجموعات جزئية من البيانات،يؤدي إلى إسناد معرف وحيد ومفرد لكل record/row سجل/صف في كل جدول (مفتاح رئيسي *primary key* a)وعمل العديد من التعديلات الأخرى،ويمكن أن يتم تطبيع البيانات من أجل كل من فعالية المعالجة وتكامل البيانات data integrity. فمن أجل الطلبيات البسيطة في الجدول السابق،يمكن أن يتم تطبيع البيانات ضمن ثلاث جداول منفصلة: واحد من أجل بنود أسطر الطلبيات،وجدول أخر من أجل الزبائن،وواحد أخر من أجل المنتجات كما في الجداول التالية وعلى الترتيب respectively.ففي كل جدول قمت بوضع علامة نجمة asterisk بجانب عنوان العمود الذي يتصرف كعمود مفتاح رئيسي.

Record ID *	Order ID	Customer ID	Product ID	Quantity
92231	10001	AA1	BEV01COF	3
92232	10001	AA1	BRD05RYE	1
92233	10002	BW3	BEV01COF	1
92234	10003	BW3	BEV01COF	2
92235	10004	CC1	CHP34PTO	7

Customer ID *	Customer Name
AA1	Al Albertson
BW3	Bill Williams
CC1	Chuck Charles

Product ID *	Product Name	Unit Price
BEV01COF	Coffee	3.99
BRD05RYE Rye	bread	2.68
BEV01COF	Coffee	3.99
CHP34PTO	Potato chips	0.99

فلضم البيانات من جداول متعددة مباشرةً،اربط(أوصل) الحقول المتطابقة.على سبيل المثال،بإمكانك وصل حقل Customer ID في جدول البنود المسطرة مع حقل المفتاح الرئيسي في جدول الزبائن.حالما يتم الربط،فتفاصيل سجل بند سطر مفرد مربوط يمكن جلبه بواسطة الربط الكامل لأسم الزبون customer name. وهو نفسه بالنسبة للربط المباشر مع أي جدولين والذين لديهما حقول قابلة للربط أو الوصل.يبين الشكل التالي العلاقات الخطية بين جداول الزبون customer، والمنتج product، و الطلبية order line .



يبين الشكل ثلاث جداول مرتبطة ولكن ماتزال تعمل كأنها جدول مفرد.

لربط الجداول ببعضها،تنفذ قواعد البيانات العلائقية لغات الاستعلام *query languages* التي تسمح لك من معالجة البيانات باستخدام الجبر العلائقي*relational algebra* (والذي اشتق منه مصطلح قاعدة البيانات العلائقية)،والأكثر شيوعاً لهذه اللغات هي ،السكول SQL،والتي تستخدم اللغة الانكليزية البسيطة مثل الجمل للربط،والترتيب،التلخيص،واستخلاص قيم البيانات التي تحتاجها تماماً.والعبارة الرئيسية،هي "اختر SELECT "،والتي توفر اختيار بيانات قاعدية واستخلاص الميزات.ويوجد أيضاً ثلاث عبارات أخرى مشتركة وهي،إدراجINSERT ،تحديثUPDATE،وحذفDELETE ,،تسمح لك هذه العبارات من معالجة السجلات المخزنة في كل جدول،فمع بعضها كل من هذه العبارات الأربع تكّون :أوامر لغة معالجة البيانات (DML) *Data Manipulation Language* الرئيسية لسكولSQL .

تحتوي سكول أيضاً على عبارات لغة تعريف البيانات(DDL) *data definition language* والتي تسمح لك تصميم الجداول المستخدمة لحفظ البيانات، بالإضافة إلى ميزات قاعدة البيانات الأخرى.وسأريك أمثلة عن عبارات سكول المتنوعة فيما بعد في هذا الفصل.

تبسط الأنظمة الخاصة للبيع Vendor-specific systems مثل خادم ميكروسوفت سكول Microsoft's SQL Server،وأوراكل Oracle's Oracle،وميكروسوفت أكسس Microsoft's Access الميزات الجوهرية لكل من DDLو DMLمن خلال تحليل إضافي للبيانات وأدواتها الإدارية.

وتكافح هذه الأنظمة مرةً أخرى على ميزات هامة مثل تكرار البياناتdata replication ،تكامل البيانات المقاومة للانهيار crash-proof data integrity،السرعة التي تعمل بها الاستعلامات المعقدة على استرجاع النتائج المطلوبة،ومن لديه أكبر تدفق خاص.

سكول سرفر 2005 . SQL Server 2005

إن أداة قاعدة البيانات الرئيسية على مستوى العمل لميكروسوفت هي سكول سرفرSQL Server .وهي على خلاف ميكروسوفت أكسس(منتج أخر معروف لقواعد البيانات العلائقية من إصدارات ميكروسوفت والذي يأتي مع مجموعة أوفيس)،يتضمن سكول سرفر إدارة بيانات وميزات تحليل متقدمة.

إن الصفات الغالبة الآن على جميع إصدارات الفيجوال أستوديو احتواءها نسخة ما من سكول سرفر،حتى تلك الإصدارات المنخفضة(المتمات).فبإمكانك استخدام سكول سرفر ضمن تطبيقات الدوت نت الخاصة بك،ولكن سكول سرفر2005 اليوم يتيح لك نحت(صناعة) الإجراءات المخزنة المضمنة باستخدام كود الدوت نت،على طول مع لغة المعالجة الإنشائية T-SQL scripting language الأصلية التقليدية.

اسم السكول سرفر SQL Server يدل ضمنياً كمنتج "خادمserver ".فهو يعمل في خلفية النظام،ويتصل معك أنت المستخدم ،فقط بامتلاكك على شبكة عمل network قياسية يتم تأسيسها لأول مرة first establish متصلة مع محرك الخدمة server engine.وهذا صحيح حتى ولو كان محرك خادم سكول مشغل على جهازك الخاص.فترقُب منتج السرفر مثير كما هو الحال بالنسبة لقراءة كتب التدريس المتخصصة tutorial books بالفيجوال بيسك2008 الأخرى والتي تتجنبها على نحو واسع،لذلك فإن ميكروسوفت توفر أدوات عميل متنوعة تتيح لك إدارة قواعد البيانات،الجداول،وخاصيات قاعدة البيانات الأخرى.بالنسبة لي لم أثبت أي من إصدارات السكول ما عدا الأدوات التركيبية التي يتم تنصيبها مع نسخة الفيجوال أستوديو2008 الاحترافية.

على الرغم من ذلك ماتزال ميكروسوفت مستمرة في تحديث وبيع ميكروسوفت أكسس،ومن الموصى به أكثر وأكثر من أجل المطورين المحترفين استخدام وتوزيع قواعد البيانات في هيئة مخدم سكول.وحتى أن ميكروسوفت ترخص لك إعادة توزيع سكول سيرفر2005 بطبعته السريعة مع تطبيقك. ولعمل هذا،عليك أولاً امتلاك "ترخيص سكول سرفر2005 بطبعته السريعة والقابلة لإعادة التوزيع" من ميكروسوفت.ولحسن الحظ فهي مجانية ويمكنك امتلاكها من موقع الانترنت التالي:

http://www.microsoft.com/sql/express

سكول (لغة الاستعلام الهيكلية)structured query language(SQL)

تولي العمل في اليابان سهل جداً،بمجرد إتقانك اللغة.ونفس الكلام صحيح فيما يخص خادم سكول:فهو سهل جداً من أجل معالجة وإمكانية الوصول للبيانات، حالما تتقن اللغة .ولكن في هذه الحالة ،اللغة هي سكولSQL ،أو لغة الاستعلام الهيكلية *Structured Query Language*.فلقد أصبحت سكول المعبر القياسي في صناعة قاعدة البيانات.

يشرح هذا الفصل عبارات DDLو DMLوالتي هي أكثر منفعة بالنسبة لتطوير برنامج المكتبة.وستكون مسرور لمعرفة أن سكول ليست بهذا الحرص pickyالكبير فيما يتعلق بهيئة العبارات المتنوعة.حيث أن الفروقات في حالة الأحرف الكبيرة والصغيرة Upper- and lowercase distinctions يتم تجاهلها،فـ SELECTهي نفس selectونفس SeLeCt.(كود T-SQL) Traditional SQL سكول التقليدي يكون على الأغلب في الحالة الكبيرة)وإني استخدم الحالة الكبيرة من أجل كل الكلمات المحجوزة،والحالة المدمجة من أجل الجداول tables،الحقولfields ،والبنود الخاصة الأخرى.

مهما يكن فإن من المهم الثبات على منوال واحد.وإن توظيف المحارف الخاصة whitespace (مثل الفراغ،أو أي محارف لاتظهر في الطباعة) يعود إليك وكما تراه مناسب.فيمكنك وضع عبارات في سطر واحد طويل .أو وضع كل كلمة في سطر مفرد.الحالة الوحيدة التي تكون فيها المحارف الخاصة وحالة الأحرف(كبيرة أو صغيرة)مسألة مهمة في المتغيرات النصية للبيانات النصية الفعلية،فما تكتبه،هو ذلك الذي يبقى.تنتهي عبارات سكول عادة بالفاصلة المنقوطة semicolon،ولكن بعض الأدوات لاتتطلب إدخال الفاصلة المنقوطةsemicolon ،وأدوات أخرى تتطلب أن تضمنها.عندما تستخدم أدوات العميل المرئية لسكول سرفر(إدارة أستوديو وإدارة أستوديو السريع)فإن الفاصلة المنقوطة اختيارية،ولكن من الجيد تضمينها عندما تستخدم عدة عبارات مع بعضها البعض،عبارة بعد عبارة.وعبارات سكول المستخدمة في كود الفيجوال بيسك لا تتضمن أبداً الفواصل المنقوطة.فيما بعد عندما تنظر إلى صيغ سكول SQL scriptالتي كتبتها،ستلاحظ الكلمة GOمرة بعد مرة،في السكول سرفر،هذا الأمر يقول"من أجل جميع العبارات الأخرى التي تظهر حتى الآن،تقدم للأمام وعالجها الآن "For all of the other statements that "appeared so far, go ahead and process them now."

عبارات "لغة تعريف البيانات" DDL Statements

من المحتمل أن تكون هذه صدمة لك،ولكن قبل أن تخزن أي بيانات في جدول،عليك أن تعمل على إنشاء ذلك الجدول.سكول لديها الأداة التامة لعمل هذا:عبارة "إنشاء جدولCREATE TABLE ".فهي عبارة من العديد من العبارات .والقواعد الأساسية basic syntax بسيطة جداً:

```
CREATE  TABLE  tableName
```

```
(
fieldName1 dataType options,
fieldName2 dataType options,
and so on...
)
```

فقط اعمل على ملئ الأجزاء وستكون جاهز لتعبئة (البيانات ،وهذا كل مافي الأمر).أسماء الحقول والجدول يتم بناءها من الحروف والأرقام،بإمكانك تضمين فراغات وبعض المحارف الخاصة،ولكن يتم عمل هذا من أجل كتابة كود فقط فيما بعد سترى.فكل بائع لديه مجمع خاص من أنواع لبيانات *data types* ، سأقتحم إصدارات سكول سرفر هنا.يسمح لك *options* من تعيين أشياء مثل فيما إذا يتطلب الحقل بيانات data أم لا،وفيما إذا يمثل المفتاح الرئيسي للجدول أم لا،وقيود *constraints* أخرى مشابهة.الامتدادات Extensions للتركيب تتيح لك تثبيت القيود constraints التي تطبق على كامل الجدول،الفهارس *indexes* (التي تسمح لك ترتيب أو البحث بعمود معين بسرعة أكبر)،وتفاصيل تخزين البيانات.

إليك مثال عن عبارة "إنشاء جدول" والتي يمكن استخدامها لجدول بنود اسطر الطلبية order line .

```
CREATE TABLE LineItems
(
RecordID bigint IDENTITY PRIMARY KEY,
OrderID bigint NOT NULL,
CustomerID varchar(20) NOT NULL
REFERENCES Customers (CustomerID),
ProductID varchar(20) NOT NULL,
Quantity smallint NOT NULL
)
```

الكلمة المحجوزة IDENTITY تسمح لخادم سكول SQL Server من تولي مسؤولية تعبئة حقل RecordID بالبيانات،وسوف تستخدم العداد التتابعي sequential counter لتزويد قيمة مميزة ومفردة لـ RecordID بالنسبة لكل سجل جديد.التعبير "مفتاح رئيسي PRIMARY KEY " يعرف الحقل RecordID كقيمة تعريف مفردة ومميزة بالنسبة لكل سجل في الجدول.أنواع البيانات bigint و smallint تشير إلى الحقول التي تحفظ قيم عددية ذات حجم مناسب،والنوع varchar يوفر مساحة لأجل النصوص،بالطول الأعظمي المخصص في الأقواس(20 characters).يعرف التعبير REFERENCES العلاقات بين جدول LineItems وجدول أخر مسمى Customers،تطابق القيم في حقل LineItems.CustomerID القيم المفتاحية من حقل Customers.CustomerID. (لاحظ تركيب "النقطة dot " لفصل أسماء الجداول والحقول.وهي تظهر في كل مكان في السكول)المرجعيات(العلاقات References) بين الجداول تعرف أيضاً علاقات(مرجعيات) أجنبية(أو ما يطلق عليه مفتاح ثانوي) *foreign references* .

إذا كنت تريد جعل هيكل أو خيار التغيرات لجدول أو حقوله بعد أن يتم إنشاءه،تتضمن سكول عبارة **ALTER TABLE** والتي يمكنها تغير تقريباً كل شيء في الجدول.ويوجد أيضاً عبارة **DROP TABLE** التي تستخدم للتخلص من الجدول وكل بياناته.ومن المحتمل انك تتجنب هذه العبارة على البيانات الموجودة في المنتج،وكمستخدم تميل للغضب قليلاً عندما تختفي فجأةً جميع بياناتك.

يلخص الجدول التالي أنواع البيانات المتاحة في مخدم سكول SQL Server.

Data type نوع البيانات	Description الشرح
Bigint	8-بايت(64-بت)لحقل عددي صحيح للقيم التي تتراوح من–9,223,372,036,854,775,808 إلى9,223,372,036,854,775,807
binary	بيانات ثنائية ثابتة الطول،حتى 8,000بايت في الطول.وإنك تخصص الطول من خلال الوسيط، مثل في binary(100)
Bit	تدعم ثلاث قيم ممكنة:1,0،أو بدون قيمة NULL.بشكل عام تستخدم للقيم المنطقيةBoolean. ذاتياً،يخزن السكول سرفر عدة حقول بت لسجل مفرد في حقل عددي صحيح مدمج.
char, nchar	(حرفيchar)قياسي ثابت الطول أو نصوص(س.حرف nchar)ليوني كودUnicode ،حتى 8,000حرف في الطول.وإنك تخصص الطول من خلال الوسيط في القوس مثلchar(100)
Cursor	يستخدم هذا النوع من البيانات ضمن الإجراءات المخزنةstored procedures ،ولا يتم استخدامه لإنشاء عمود.
Datetime	لحقول الوقت والتاريخ العام التي تتراوح من January 1, 1753،إلىDecember 31, 999 ،ودقة الوقت Time accuracy لأي قيمة ثابتة ضمن 3.33ميلي ثانية.يعمل السكول سرفر2008على إضافة العديد من البيانات المتعلقة بأنواع البيانات: date(تواريخ دون وقت)،وقت time(أوقات دون تواريخ)،تاريخ ووقت2 datetime2 (نفس الحالي datetime،ولكن بمجال أعلى ودقة إلى 100 بليون من الثواني100 nanoseconds)، و datetimeoffset (مجالات للوقت والتاريخ)
decimal, numeric	حقل بتدريج عشري وثابت الدقة.إنك تحدد العدد الأكبر من الأرقام التي تظهر على جانبي النقطة العشرية (الدقةprecision) والعدد الأكبر لهذه الأرقام التي يمكن أن تظهر على الجهة اليمنى للنقطة أو الفاصلة العشرية decimal point (التدريجscale).على سبيل المثال،وضع (decimal(10,4يعمل على إنشاء حقل حتى عشرة أرقام بالكامل،وأربع منها يمكن أن تظهر بعد الفاصلة أو النقطة العشرية.وقيمة الدقة الأعلى هي 38.و numericمرادف synonym لـdecimal،مثل dec.
Float	حقل النقطة العشرية العائمة(الكسرية)مع التخزين المتغير.بإمكانك تخصيص عدد الوحدات(البتاتbits)المستخدمة لتخزين قيمة،حتى 53.بشكل افتراضي،جميع الوحدات ال53 يتم استخدامها،لذلك وضع float مكافئ لـfloat(53)

	.ونوع البيانات السابق realمكافئ لـfloat(24). والقيم المخزنة على هيئة±1.0×10±38 ، تفاوت الدقة ونفس المجال بالوحدات bitsالمستخدمة لتخزين نوع البيانات هذه حساس لأقل عدد من الأخطاء المحسوبة.
Hierarchyid	نوع البيانات هذا جديد،في السكول سرفر2008،ويدعم الاستعلام الهرمي(التسلسلي)والبيانات الشجرية الشكل(الهيئة tree-shaped data).وهي غير متاحة في سكول سرفر2005.
image, text, ntext	لاتستخدم أنواع البيانات هذه ،حيث أنه سيتم إزالتها في النهاية من سكول سرفر.
Int	حقل عددي صحيح 4-بايت(32-بت) للقيم ضمن المجال من–2,147,483,648 إلى2,147,483,647 .
Money	حقل عالي الدقة high-accuracy 8بايت(64بت)لتخزين القيم النقدية(عملةcurrency values)،حتى أربع أرقام للفاصلة العشرية(النقطة العشرية).ويخزن مجال لقيم البيانات من–922,337,203,685,477.5808 إلى 922,337,203,685,477.5807 .
rowversion,timestamp	يتم استخدام نوع البيانات هذه لتسجيل أحداث التعديل المطبقة على السجلات.ويوجد قيود على استخدامها،وهي غير مضمونة لأن تكون مفردة ومميزة ضمن الجدول. timestampهو مكافئ قليل الأهمية deprecated synonym لـ rowversion،استخدم rowversionبدلاً عنه.
Smalldatetime	حقل للوقت والتاريخ العام للتواريخ ذات المجال من January 1, 1900 AD إلى June 6, 2079 AD. ودقة الوقت Time accuracy لأي قيمة ثابتة هي ضمن دقيقة واحدة.
Smallint	حقل(متغير)عددي صحيح 2بايت(16بت) للقيم التي تتراوح من –32,768 إلى 32,767 .
Smallmoney	حقل عالي الدقة4بايت(32بت)لتخزين القيم النقدية(عملة)،حتى أربع أرقام بعد الفاصلة العشرية.وقيم البيانات المخزنة تتراوح من–214,748.3648 إلى 214,748.3647 .
sql_variant	نوع عام يخزن القيم للعديد من أنواع الحقول المخصصة الأخرى.
Table	حقل خاص يخزن بشكل مؤقت نتائج الاستعلام على هيئة جدول مضغوط.وتعريف حقل(متغير)جدول معقد نوعاً ما،ويتم استخدامه عادة مصحوباً بقيود خاصة.
Tinyint	حقل عددي صحيح دون إشارة unsigned integer field 1بايت(8بت) لمجال القيم من 0إلى 255.
Uniqueidentifier	معرف مفرد ومميز شامل(globally unique identifier (GUID 16بايت .تولد الدالة(الوظيفة) NEWIDقيم لهذا الحقل أو المتغير.
Varbinary	بيانات ثنائية متغيرة الطول،حتى 8000بايت في الطول.إنك تحدد الطول من خلال الوسيطparameter ، مثل varbinary(100).ويأخذ المتغير(الحقل) المساحة من أجل المحتوى الفعلي الحالي المخزن في الحقل.وإن الإعداد الخاص له مثل varbinary(max) يتيح مدخلات حتى 2بليون بايت تقريباً.
varchar, nvarchar	متغير ذو طول قياسي(varchar)أو نصوص يوني كود Unicode (nvarchar)،حتى 8000 حرف في الطول،وإنك تحدد الطول من خلال الوسيط مثل varchar(100) . ويأخذ الحقل مساحة فقط لما تم تخزينه في الحقل. والإعداد الخاص له varchar(max) يسمح مدخلات حتى 2بليون حرف تقريباً.
Xml	توفر التخزين لمستندات(وثائق)بيانات XMLالمرمزة typedوالغير مرمزة untyped ،حتى 2غيغابايت2 GB.

عبارات "لغة معالجة البيانات"DML Statements

تضيف العبارة INSERTسجلات(صفوفrecords)بيانات للجدول .يتم إضافة البيانات Dataللجدول سجل واحد كل مرة .(تنوع عبارة INSERT يتيح لك إدخال عدة سجلات ،ولكن هذه السجلات يجب أن تأتي من مصدر آخر لجدول موجود) لاستخدام عبارة INSERT،حدد الجدول والحقول المقصودة ومن ثم القيم المستقلة التي يتم وضعها في كل حقل .كل قيمة للبيانات تتوافق مع كل اسم عمود بيانات محدد.

```
INSERT INTO LineItems
(OrderID, CustomerID, ProductID, Quantity)
VALUES    (10002, 'BW3', 'BEV01COF', 1)
```

خذ هذه العبارة بالمقارنة مع عبارة CREATE TABLE المكتوبة سابقا ،سيعمل إجراء الإدخال هذا على إضافة سجل جديد لجدول LineItemsمع خمسة حقول (أو أعمدة)جديدة-أربع حقول معينة ،زائد حقل المفتاح الرئيسي primary key الذي يتم إضافته بشكل آلي لحقل RecordID (حيث أنه تم تعليمه كمعرف IDENTITY).يعمل سكول سرفر اختبار تكامل البيانات المتنوعة بالنيابة عنك.كل حقل بيانات تعمل على إضافته يجب أن يكون من نوع البيانات الصحيح.بما أننا قمنا بتصميم حقل CustomerIDعلى علاقة مع جدول Customer،سيفشل الإدخال إذا كان BW3غير موجود سابقاً في جدول Customer.

الحرفية العددية يمكن أن يتم تضمينها في عبارات سكول عند الحاجة دون أي شرط أو قيد إضافي.الحرفية النصية هي دائماً تكون محاطة بعلامة تنصيص(اقتباس) مفرد،كما تم عمله لمعرفات حقول customer وproductفي عبارة الإدخال INSERTهذه.وإذا أردت أن تضمن علامة تنصيص مفردة في الحرفية،قم بإدخاله مرتين:

```
'John O''Sullivan'
```

قم بإحاطة البيانات الحرفية literalوالقيم الزمنية time values بعلامات تنصيص مفردة:

```
'7-Nov-2005'
```

مثل قيم التاريخ والوقت هذه تقبل أي تنسيق مميز،على الرغم من أنك ستستخدم تنسيق ليس من السهل على سكول سرفر أن يسيء تفسيرهmisinterpret . العديد من أنواع الحقول تدعم قيمة "غير مسندةunassigned "،قيمة تشير أن الحقل لا يحتوي بيانات على الإطلاق.مثل هذه القيم تعرف "بدون قيمة null "،ويتم تخصيصها في سكول سرفر باستخدام الكلمة المحجوزة "بدون قيمة NULL ".ولا يمكنك إسناد قيمة "بدون قيمة NULL " لحقل المفتاح الرئيسي،أو لأي حقل تم تعليمه بالخيار NOT NULL.

لإزالة سجل تم إضافتها سابقاً،استخدم عبارة DELETE:

```
DELETE FROM LineItems WHERE RecordID = 92231
```

تتضمن عبارة DELETEالشرط(أو التعبيرWHERE حيث)(الجزء WHERE RecordID = 92231).يسمح لك الشرط WHEREمن الإشارة إلى سجل أو أكثر في جدول ما وذلك بعمل مقارنة مع بيانات الحقول.ويمكن لشرط WHEREأن يتضمن كلمات محجوزة ANDو ORلربط عدة شروط،وأقواس parenthesesللتجميع.

```
DELETE FROM LineItems WHERE OrderID = 10001
AND ProductID = 'BRD05RYE'
```

يمكن لمثل عبارة DELETEهذه أن تحذف 0سجل،سجل واحد أو1000سجل،ولذلك فإن الدقة في شرط WHEREهام جداً.لحذف جميع السجلات في الجدول قم بإخراج شرط بالكاملWHERE .

```
DELETE FROM LineItems
```

تستخدم عبارة "تحديث UPDATE " أيضاً تعبير WHEREلتعديل القيم في سجلات جدول موجود.

```
UPDATE    LineItems
SET           Quantity = 4
WHERE     (RecordID = 92231)
```

يتم عمل الإسنادات للحقول بالتعبير "ضع SET "،ضع اسم الحقل(Quantity)على الجهة اليسارية من إشارة المساواة ،والقيمة الجديدة على الجهة اليمنية من إشارة المساواة(4).ولإسناد عدة قيم في نفس الوقت،افصل كل إسناد بالفاصلة comma.وبإمكانك أيضاً تضمين صيغ formulas وحساباتcalculations .

```
UPDATE LineItems SET Quantity = Quantity + 1,
ProductID = 'BEV02POP'
WHERE RecordID = 92231
```

وكما مع عبارة DELETE،فيمكن لعبارة UPDATEأن تحدث صفر سجل،سجل وحيد،أو عدة سجلات بالاعتماد على أي سجلات تتطابق مع الشرط WHERE.

العبارة الأخيرة للغة معالجة البيانات DML،والأكثر استخداماً هي عبارة "اختر SELECT ".

```
SELECT ProductID, Quantity FROM LineItems
WHERE RecordID = 92231
```

تقوم عبارة SELECTبعمل مسح على جدول(LineItems)،والبحث عن جميع السجلات التي تطابق المعيار المقدم(= RecordID 92231)،وتعود بأصغر جدول يحتوي فقط الحقول المشار إليها (ProductID، Quantity) للسجلات المتوافقة.والاستعلام الأكثر عمومية والذي يعود بكل الصفوف(السجلات) والحقول(الأعمدة) هو التالي: SELECT * FROM LineItems

يعود هذا الاستعلام بجميع السجلات من الجدول LineItems بدون أي ترتيب معين. وتعني asterisk النجمة(*) تضمين كل الحقول"include all fields." .

يعود الشرط "ترتيب بواسطةORDER BY " بنتائج مرتبة بطريقة خاصة(معينة):

```
SELECT * FROM LineItems
WHERE Quantity > 5
ORDER BY ProductID, Quantity DESC
```

يعود هذا الاستعلام بجميع السجلات التي تكون فيها قيمة حقل Quantityأكبر من 5،ويرتب النتائج أولاً بواسطة عمود ProductID (بترتيب تصاعديascending order)ومن ثم بواسطة حقل ذو القيمة العدديةnumeric quantity (بترتيب تنازلي in descending order،وذلك من خلال كتابة الكلمة DESC).

تتيح لك دوال الإجمالي (التراكم)Aggregate functions وميزات التجميع grouping features تلخيص النتائج من مجموعة أكبر من البيانات.يوثق الاستعلام التالي الكمية المطلوبة الإجمالية total ordered quantity لكل منتج product في الجدول:

```
SELECT ProductID, SUM(Quantity) FROM LineItems
GROUP BY ProductID
```

بإمكانك استخدام *joins*لربط البيانات مع بعضها من جدولين مميزين أو أكثر.الاستعلام التالي يضم جدول LineItemsوجدول Customerعلى أعمدتهم CustomerID المتطابقة. وتوضح أيضاً عبارة SELECTهذه استخدام اختصارات abbreviations الجدول(السابقات LI و CU)المضافة من خلال التعبير AS،وهي ليست ضرورية عادةً،ولكنها تساعد على جعل الاستعلام المعقد أكثر قابلية للقراءة:

```
SELECT LI.OrderID, CU.CustomerName, LI.ProductID
FROM LineItems AS LI INNER JOIN Customer AS CU
ON LI.CustomerID = CU.CustomerID
ORDER BY LI.OrderID, CU.CustomerName
```

يستخدم هذا الجدول "الربط الداخليinner join "وهو واحد من خمسة أنواع للربط،وكل منها يعود بمجموعات مختلفة من السجلات بالاعتماد على العلاقة بين الجدول الأول(اليساري) والجدول الثاني(اليميني)في الربط join.

الربط الداخلي: *Inner join*

يعود فقط بالسجلات التي تتطابق مع بعضها في الحقول المربوطة.وهذا النوع من الربط يستخدم الكلمة المحجوزة INNER JOIN.

الربط الخارجي اليساري : *Left outer join*

يعود بكل سجل من الجدول اليساري وفقط بالسجلات المطابقة من الجدول في الجهة اليمنية من الربط،فإذا كان سجل الجدول اليساري ليس لديه تطابق،فإنه يتصرف وكأن جميع حقول الجدول اليميني لذلك السجل تحتوي قيم "بدون قيمة NULL ".هذا النوع من الربط يستخدم الكلمة المحجوزة LEFT JOIN.وأحد استخداماته يمكن أن يكون لربط جدول Productوجدول LineItems. فبإمكانك إرجاع قائمة بكامل أسماء المنتجات لكل المنتجات المتاحة ،بالإضافة إلى الكمية الإجمالية المطلوبة من كل منتج.

بوضع جدول المنتجات Productعلى الجهة اليسارية من الربط الخارجي اليساريleft outer join ،فإن الاستعلام سيعود بأسماء كل المنتجات،حتى ولو كان ذلك المنتج لم يتم طلبه على الإطلاق(ولا يظهر في جدول LineItems).

الربط الخارجي اليميني *Right outer join*

يعمل هذا تماماً مثل الربط الخارجي اليساري،ولكن جميع سجلات الجدول اليميني يتم استعادتها،واستعادة فقط سجلات الجدول اليساري التي لديها تطابق. ويستخدم هذا النوع من الربط الكلمة المحجوزة RIGHT JOIN.

الربط الخارجي التام *Full outer join*

يعود بجميع السجلات من الجدولين اليساري واليميني،فيما إذا كان هناك تطابق أم لا.عندما يوجد تطابق،فإنه يتم عكس النتائج it is reflected in the results.وهذا النوع يستخدم الكلمة المحجوزة FULL JOIN.

الربط المتقاطع(المتصالب)*Cross join*

ويدعى أيضاً الربط الديكارتي*Cartesian join* (دمج مجموعتين لتوليد مجموعة واحدة مدمجة).ويعود بكل السجلات المدمجة الممكنة اليمنية واليسارية.ويستخدم هذا النوع الكلمة المحجوزة CROSS JOIN.

يركز الربط على العلاقة التي بين الجدولين.(وهذا الاستخدام لـ"العلاقة" ليس الأساس لمصطلح قاعدة البيانات العلائقية *relational database*)بعض الجداول توجد في علاقة "أباء- أبناءparent-child "،فسجل واحد"أب"لديه واحد أو أكثر من السجلات الأبناء المعتمدة في جدول أخر.وهذا غالباً صحيح للطلبياتorders ،فعنوان طلبيةorder header "مفرد لديه عدة "بنود أسطرline items "وهذا النوع من العلاقة يعرف بواحد-إلى –العديد *one-to-many* ،حيث أن سجل واحد يكون مرتبط بعدة سجلات في جدول أخر.والعلاقة تكون أحادية الاتجاه،والسجل الابن المعطى لايرتبط بسجلات أبوية متعددة.

تربط العلاقة واحد-إلى –واحد*one-to-one* سجل وحيد في جدول واحد إلى سجل في جدول أخر.وهي بسيطة جداً،وتستخدم غالباً لتحسين القيم الموجودة في السجل الأصل من السجل التكميلي في الجدول الثاني.

في علاقة عديد – إلى –عديد*many-to-many* ، سجل مفرد في جدول واحد يكون مصحوباً مع عدة سجلات في الجدول الثاني،وسجل مفرد في الجدول الثاني يكون أيضاً مصحوباً بعدة سجلات في الجدول الأول.ومثال حقيقي عنه ،العلاقة بين المعلمين والطلاب في وضع كلية ما.فمعلم واحد لديه العديد من الطلبة في الصف (الحجرة الدراسية)ولكن كل طالب أيضاً لديه العديد من المعلمين في كل فصل دراسي semester.

يتطلب التنفيذ التطبيقي لعلاقة عديد-إلى –العديد عملياً ثلاث جداول:الجدولين المرتبطين ،وجدول "وسيط بينهماgo-between " والذي يربطهما مع بعض.وسأريك مثال لمثل هذا الجدول في المقطع التالي من هذا الفصل.

ما خلف أساس سكول Beyond Basic SQL

العبارات البسيطة التي جدولتها هنا تعالج موضوع إمكانيات معالجة البيانات المتاحة بشكل سطحي scratch the surface من خلال سكول.ولكن حالياً يمكن أن تكون قد لاحظت أن سكول يحاكي اللغة الانكليزية في القواعد(التركيب) على نحو رائع،وحتى أكثر مما هي عليه الفيجوال بيسك.في الحقيقة،الاسم الأصلي للغة SEQUEL والذي كان تركيب لـ"لغة الاستعلام الانكليزية الهيكلية Structured English Query Language " وبما أن عبارات سكول قد أخذت بالتعقيد أكثر،فستبدو أكثر شبهاً لمجموعات عشوائية من الكلمات الإنكليزية.

الهدف هنا هو تقديمك للتركيب القاعدي لعبارات سكول.معظم العبارات التي سنواجهها في مشروع المكتبة لن تكون أكثر تعقيداً من العبارات التي تم تضمينها في الأمثلة هنا.

استخدام قواعد البيانات في الفيجوال بيسك Using Databases in Visual Basic

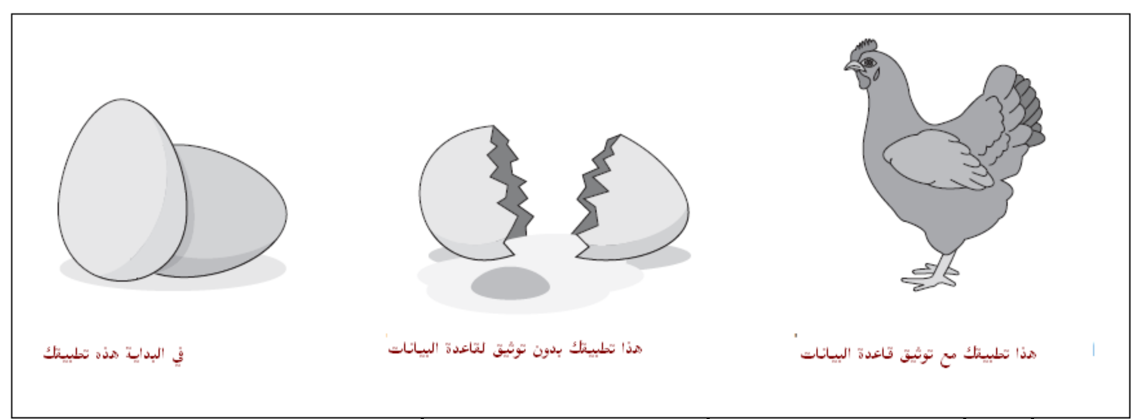
يمكن للفيجوال بيسك أن تتفاعل مع البيانات المخزنة في قاعدة بيانات بعدة طرق مختلفة:

- ✓ استخدام ADO.NET،التقنية الرئيسية للوصول للبيانات المضمنة في إطار عمل الدوت نتNET Framework. للتفاعل مع محتوى قاعدة البيانات المخزنة.وهذه هي الطريقة المستخدمة على طول مع برنامج المكتبة للتفاعل مع قاعدة بياناته. وتم شرح آدو دوت نت في الفصل العاشر،مع أمثلة عن طريقة استخدامه،وسأقدم أيضاً في ذلك الفصل كود خاص من مشروع المكتبة.
- ✓ استخدام ميزات "تحزيم البياناتdata binding "المتاحة في الفيجوال بيسك والفيجوال أستوديو.يؤسس التحزيم اتصال بين أداة البيانات التي على الشاشة أو كائن تمكين البيانات المشابه والمحتوى من قاعدة بيانات ما.الكود المكتوب لك بواسطة ميكروسوفت يأخذ على عاتقه عمل كل الاتصالات،فتستطيع حتى سحب و إسقاط هذه الأنواع من التفاعلات .على الرغم من أنني سأناقش تحزيم البيانات في الفصل العاشر(حيث أن التحزيم يعتمد علىADO.NET)،فإني أميل لتجنبه،لأنه يقلل مقدار تحكم المبرمج الذي يستطيع أن يبذله على معالجة بيانات المستخدم.ولن يتم استخدام تحزيم البيانات في برنامج المكتبة.
- ✓ استخراج البيانات من قاعدة البيانات ضمن ملف قياسي،واستخدام ميزات معالجة ملف في الفيجوال بيسك لمعالجة البيانات.من الواضح أن هذا غير مفيد جداً،ولكنني مضطر لاستخدامه،خاصة في الأيام القديمة عندما كانت قواعد البيانات الخاصة لاتستطيع التفاعل بسهولة مع كود الفيجوال بيسك.
- ✓ كل مرة تحتاج فيها لبعض البيانات،اخبر المستخدم أنه بطريقة أو بأخرى قد تم فقدها،وبالتالي يتوجب إعادة إدخالها مباشرةً. فإذا كان لديك الفضول لمعرفة كيف يبدو داخل مكتب البطالة،فهذه يمكن أن تكون فرصتك.

توثيق قاعدة البيانات Documenting the Database

المحتوى التقني الذي يشرح الجداول والحقول في قاعدة بياناتك للتطبيق تمثل معظم الجزء الهام من التوثيق المولد خلال حياة التطبيق.في الواقع،الحاجة لتوثيق جيد هو الأساس فيما يخص واحد من معتقداتي البرمجية الجوهرية:توثيق المشروع هو هام،وفي بعض الأحيان أكثر أهمية حتى من الكود المصدريsource code .

إذا كنت تطور تطبيق ما يتمحور حول قاعدة بيانات تخزن محتوى لمستخدم،التوثيق الكامل والدقيق لكل جدول وحقل مستخدم في قاعدة البيانات هو واجب.وأي نقص في هذه المساحة يجب أن لايكون ممكن أو محتمل،وإذا كان فإنه سيقود قضايا تكامل البيانات وأبعد من التسلسل الزمني للتطور الضروري كما هو مبين في الشكل التالي:



السبب الذي يجعلني أعتقد أن توثيق قاعدة البيانات أهم من توثيق المستخدم أو المواصفات الوظيفية،هو الأثر الذي سيتركه المستند على بيانات المستخدم. فإذا كان لديك قاعدة بيانات موثقة،فتستطيع أن تخمن ماهية المواصفات الوظيفية،وربما تأتي هذه التخمينات دقيقة جداً.فإذا كان ينقصك توثيق المستخدمuser documentation ،بإمكانك دائما كتابته عندما ينتهي البرنامج .ولكن إذا كان ينقصك توثيق قاعدة البياناتdatabase documentation ،فأنت داخل عالم مؤذٍ.

إذا لم تعمل من قبل على مشاريع قاعدة بيانات ضخمة،من المحتمل أن لا تصدقني،ولكن لو عملت على وراثة نظام قاعدة بيانات واسع ومتكامل ويحوي على الكثير من الجداول وقد شارك فيه أكثر من مطور وكل منهم وضع مجال مختلف من البيانات الذي سيسمح لكل حقل أن يحتويها،وأي من الحقول يجب أن تكون مطلوبة أم لا ،فهنا سترى التضارب.

فتتبع 100,000 سطر من الكود المصدري لتحديد ما يعمله كل حقل(متغير) ليست بالتسلية،وقد تأخذ هذه العملية عدة شهور لإتمامها بدقة.لذلك لا تدع هذا يحصل معك.

المشروع

لنبدأ بمجموعة المصادر التقنية لمشروع المكتبة وذلك بتصميم وتوثيق جداول قاعدة البيانات لكي يتم استخدامها بواسطة التطبيق.وتظهر مجموعة المصادر هذه في دليل تثبيت الكتاب،في الدليل الجزئي للفصل الرابع،وتحتوي على الملفات التالية:
نسخة من ميكروسوفت ورد لتوثيق المشروع تقنياً وهي مقدمة في هذا المقطع مشروع هذا الفصل (كامل التوثيق موضوع هنا).

سكول.صيغة إنشاء قاعدة البيانات*Database Creation Script.sql*

صيغة قاعدة بيانات سكول سرفر SQL Server database script المستخدمة لبناء الجداول والحقول الفعلية في قاعدة البيانات.

محتوى مجموعة المصدر التقني Technical Resource Kit Content

يحتوي هذا المقطع جدولة بالجداول المحتواة في قاعدة بيانات المكتبة.كل جدول يتضمن وصف عام لمساعدتك في فهم هيكل قاعدة البيانات.وسوف تصادف جميع الجداول في الفصول اللاحقة،على طول مصحوبة بالكود المصدريsource code ،لذلك لا تغضب freak out إذا بدت بعض الجداول أو الحقول صعبة الفهم لك الآن.

جداول الأمن المترابطة Security-related tables

على الرغم من أن الزبائن patrons ليسوا بحاجة لتسجيل الدخولlog in للتطبيق لاستخراج البنودlook up من قاعدة البيانات،فالمدراءadministrators يجب أن يسجلوا الدخول قبل أن يتمكنوا من الوصول إلى الميزات المحسنة للبرنامج.فالجداول الأربع التالية تدير إعتمادات الأمن security credentials لكل مدير.يستخدم التطبيق قاعدة بيانات سكول سرفر أوWindows-based security credentials إعتمادات التأمين المعتمدة على ويندوز فقط للوصول إلى قاعدة البيانات بشكل أولي,initially،وليس للحد من(تقيدrestrict) الميزات.

النشاط **Activity**.يعرف هذا الجدول ميزات التطبيق التي يمكن تأمينها باستخدام مجموعة الحقوقgroup rights .وهذه النشاطات يتم ربطها مع مجموعات الأمن(من جدول GroupName) لتأسيس الحقوق لمجموعة خاصةparticular group .

Description	Type	Field
مفتاح رئيسي.وهذا المفتاح آلي التوليد،والقيمة المزودة تطابق داخلياً القيم المستخدمة ضمن مشروع المكتبة.وهو مطلوب.	Long	ID
Descriptive name of this activity. Required. اسم وصفي لهذا النشاط.مطلوب	Text(50)	FullName

حالياً يتم تعريف النشاطات التالية:

- 1 - Manage authors and names إدارة المؤلفين والأسماء
- 2 - Manage author and name types إدارة المؤلف وأنواع الأسماء
- 3 - Manage copy status codes إدارة أكواد الحالة للنسخ
- 4 - Manage media types إدارة أنواع الوسائط
- 5 - Manage series إدارة السلاسل
- 6 - Manage security groups إدارة مجموعات الأمن
- 7 - Manage library materials إدارة مواد المكتبة
- 8 - Manage patrons إدارة الزبائن
- 9 - Manage publishers إدارة الناشرين
- 10 - Manage system values إدارة قيم النظام

- 11 - Manage administrative users إدارة المستخدمين الإداريين
- 12 - Process and accept fees معالجة وقبول الرسوم
- 13 - Manage locations إدارة المواقع
- 14 - Check out library items تفحص بنود مخرجات المكتبة
- 15 - Check in library items تفحص بنود مدخلات المكتبة
- 16 - Access administrative features الميزات الإدارية للوصول
- 17 - Perform daily processing عمل معالجة يومية
- 18 - Run system reports تشغيل تقارير النظام
- 19 - Access patrons without patron password تمكين الزبائن من الوصول دون الحاجة لكلمة مرور الزبون
- 20 - Manage barcodes إدارة أكواد التعريف
- 21 - Manage holidays إدارة أيام العطل
- 22 - Manage patron groups إدارة مجموعا ت الزبائن
- 23 - View administrative patron messages عرض رسائل الزبائن الإدارية

أسماء المجموعات **GroupName**. كل سجل في هذا الجدول يعرف مجموعة أمن مفردة.فأمناء المكتبة Librarians والمدراء administrators الأخريين كل منهم ينتمي إلى مجموعة أمن مفردة.

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required. مفتاح رئيسي،يتم إسناده بشكل آلي.وهو مطلوب
FullName	Text(50)	Name of this group. Required. اسم هذه المجموعة. مطلوب

نشاط مجموعة **GroupActivity**. هذا الجدول يصل السجلات في جدول النشاط Activityإلى السجلات في جدول اسم المجموعة GroupName(علاقة عديد-إلى-عديدmany-to-many) لتأسيس نشاطات مجموعة أمن التي يمكن عملها.

Field	Type	Description
GroupID	Long	Primary key. The associated security group. Foreign reference to GroupName.ID. Required. مفتاح رئيسي.المصاحب لمجموعة تأمين.مفتاح ثانوي(علاقة ثانوية) لحقل معرف اسم مجموعة GroupName.ID .مطلوب
ActivityID	Long	Primary key. The activity that members of the associated security group can perform. Foreign reference to Activity.ID. Required. مفتاح رئيسي. النشاط الذي تستطيع عمله الأعضاء المصاحبة لمجموعة تأمين. مفتاح ثانوي لحقل النشاط Activity. ID.مطلوب .

اسم المستخدم **UserName**.يحتوي هذا الجدول السجلات الحقيقية لكل أمين مكتبة أو مدير.يتضمن كل سجل كلمة مرور المستخدم ووضع مجموعة الأمن .

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required. مفتاح رئيسي،إسناد آلي. مطلوب
FullName	Text(50)	Name of this user, administrator, or librarian. Required. اسم المستخدم،المدير،أو أمين المكتبة. مطلوب .
LoginID	Text(20)	User ID that gives this user access to the system. It is entered into the Library program's "login" form, along with the password, to gain access to enhanced features. Required. معرف المستخدم والذي يسمح للمستخدم من الوصول إلى النظام.ويتم إدخاله في برنامج المكتبة "تسجيل الدخول"على طول مع كلمة المرور،لتستطيع الدخول إلى الميزات المحسنة. مطلوب.
Password	Text(20)	The password for this user, in an encrypted format. Optional. كلمة المرور الخاصة بهذا المستخدم،بتنسيق مشفر. اختياري .
Active	Boolean	Is this user allowed to access the system? 0 for False, 1 for True. Required. هل سيسمح لهذا المستخدم من التمكن الوصول إلى النظام؟0لخطأ،1 لصواب. مطلوب .

Field	Type	Description
GroupID	Long	To which security group does this user belong? Foreign reference to GroupName.ID. Required. إلى أي مجموعة أمن ينتمي هذا المستخدم؟مفتاح ثانوي لـ GroupName.ID. مطلوب .

جداول دعم الكود Support code tables

توجد العديد من الجداول تعمل بكل بساطة على تزويد قائمة من القيم لجداول أخرى.في تطبيق ما،قائمة الجداول هذه غالباً ما تظهر كخيارات في أداة تتوسع للأسفلdrop-down ("صندوق مركبcombo box ").

نوع مؤلف الكود(أو البند) **CodeAuthorType**.في برنامج المكتبة،الكلمة *author* هي مصطلح عام مستخدم للمؤلفين authors،الرسامين،المحررين editors،وأي مساهم contributorمشابه أخر لبند ما في فاتورة المكتبةlibrary's inventory . يتيح لك هذا الجدول من تعريف هذه الأدوار(الوظائف roles).

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required. مفتاح رئيسي،إسناد آلي. مطلوب .
FullName	Text(50)	Name of this type of author or contributor. Required. اسم نوع المؤلف هذا أو المساهم. مطلوب .

وضع نسخ الكود **CodeCopyStatus**.تتضمن أكواد وضع النسخ أشياء مثل "الدوران أو معادة circulating "،"المصححة being repaired "،وأي حالة رئيسية أخرى ترغب المكتبة بوضعها.حالة تم إعادتها checked-in أو تم إخراجها أو إعارتها checked-out يتم معاملتها من خلال ميزات أخرى،كما في العلامة التي تشير فيما إذا بند ما هو بند مرجعي reference item.

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required. مفتاح رئيسي،آلي الإسناد. مطلوب .
FullName	Text(50)	Name of this status entry. Required. اسم مدخلة الحالة.مطلوب .

موقع الكود(أو البند) **CodeLocation**.المواقع الفعلية حيث يتم تخزين بنود المكتبة .وهذا ما يمكن من فصل المواقعseparate sites، أو الغرف roomsأو المساحات areasضمن الموقع المشتركcommon location .

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required.
FullName	Text(50)	Name of this location. Required.
LastProcessing	Date	The date when Daily Processing was last done for this location. If NULL, processing has not yet been done. Optional. التاريخ،عند عمل أخر معالجة يومية لهذا الموقع.إذا كان بدون قيمة،فإن المعالجة لم يتم عملها حتى الآن.اختياري.

نوع وسائط الكود **CodeMediaType**.نوع الوسائط،مثل الكتب books،المجلاتmagazines ،الفيديو videos،السيديات CDs.

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required.
FullName	Text(50)	Name of this media type. Required.
CheckoutDays	Integer	Number of days for which items in this type can be checked out, before renewal. Required. عدد أيام البنود التي يمكن استخراجها للبنود التي في هذا النوع،قبل أن يتم تجديدها.مطلوب.
RenewDays	Integer	Number of days to add to the original checkout period for a renewal of items within this type. Required. عدد الأيام التي يجب إضافتها إلى فترة الاستخراج للبنود المعاد تجديدها ضمن هذا النوع.مطلوب.
RenewTimes	Integer	Maximum number of times the item can be renewed by a patron before it must be returned. Required. العدد الأعظمي من المرات التي يمكن تجديد البند فيها من قبل الزبون قبل أن يعيده.مطلوب.
DailyFine	Currency	Amount charged per day for an overdue item of this type. Required.

Field	Type	Description
		الكمية المحسوبة على كل يوم تأخير، لبند من هذا النوع.مطلوب.

مجموعة زبائن كود **CodePatronGroup**.تصنيفات المجموعات والتي يتم وضع الزبائن ضمنها.وهي ليست مجموعات أمن،ولكن هي مجموعات عامة لأهداف تقريريةreporting purposes . وتم إضافة هذا لدعم تصنيف أو تجميع الزبائن بوحدات ضمن الشركة،أو بالفئة/الرتبة class/grade ضمن مجموعة مكتبة المدرسة .

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required.
FullName	Text(50)	Name of this patron group. Required.

CodeSeries.بعض البنود تظهر كجزء من سلاسل أكبر larger series أو تجمع collection. فيعرف هذا الجدول أسماء السلاسل والتجمع .

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required.
FullName	Text(50)	Name of this series or collection. Required.

بنود المكتبة Library items

الجداول في هذا المقطع تدير الفواتير الحقيقيةactual inventory. حيث أنه يمكن للمكتبة أن تحوز على أكثر من نسخة لبند مفرد،وتدير هذه الجداول "البند المعين بالاسمnamed item "وهي نسخ copies مستقلة وبشكل منفصل .

NamedItem. بند المكتبة،مثل كتاب ما،سيدي،أو مجلة .ويمثل هذا الجدول بند عام،وليس النسخة الفعلية للبند.

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required.
Title	Text(150)	Title of this item. Required.
Subtitle	Text(150)	Subtitle of this item. Optional.
Description	Memo	Full description of this item. Optional.
Edition	Text(10)	Edition number for this item. Optional.
Publisher	Long	This item's publisher. Foreign reference to Publisher.ID. Optional.
Dewey	Text(20)	Dewey decimal number. Use "/" for line breaks. Optional.
LC	Text(25)	Library of Congress number. Use "/" for line breaks. Optional.
ISxN	Text(20)	ISBN, ISSN, or other standardized number of this item. Optional.
LCCN	Text(12)	Library of Congress control number. Optional.
Copyright	Integer	Year of original copyright, or of believed original copyright. Optional.
Series	Long	The series or collection in which this item appears. Foreign reference to CodeSeries.ID. Optional.
MediaType	Long	The media classification of this item. Foreign reference to CodeMediaType.ID. Required.
OutOfPrint	Boolean	Is this title out of print? 0 for False, 1 for True. Required.

ItemCopy نسخة مفردة لبند معين باسمه.النسخ المنفصلة لنفس البند ستظهر كسجلات منفصلة في هذا الجدول.

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required.
ItemID	Long	The related named item record. Foreign reference to NamedItem.ID. Required.
CopyNumber	Integer	Numbered position of this item within the set of copies for a named item. Required, and unique among items with the same ItemID field value.
Description	Memo	Comments specific to this copy of the item. Optional.
Available	Boolean	Is this copy available for checkout or circulation? 0 for False, 1 for True. Required.

Description	Type	Field
Has this copy been reported missing? 0 for False, 1 for True. Required.	Boolean	Missing
Is this a reference copy? 0 for False, 1 for True. Required.	Boolean	Reference
Any comments relevant to the condition of this copy. Optional.	Text(30)	Condition
Date this copy was acquired by the library. Optional.	Date	Acquired
Value of this item, either original or replacement value. Optional.	Currency	Cost
The general status of this copy. Foreign reference to CodeCopyStatus.ID. Required.	Long	Status
Bar code found on the copy. At this time, only numeric bar codes are supported. Optional.	Text(20)	Barcode
The site or room location of this item. Foreign reference to CodeLocation.ID. Optional.	Long	Location

Publisher المنظمة التي تنشر الكتب أو بعض الأنواع الأخرى من الوسائط.

Description	Type	Field
Primary key; automatically assigned. Required.	Long - Auto	ID
Name of the publisher. Required.	Text(100)	FullName
URL for this publisher's web site. Optional.	Text(255)	WebSite

Author الشخص الذي يكتبwrites، يحررedits، يوضحillustrates، أو بطريقة أخرى يساهم contributes في بند كتاب أو بند وسيط.في جميع الحالات،عندما يظهر بند المؤلف *author*في هذا الجدول،فإنه يشير إلى أي شخص يشارك في البند .

Description	Type	Field
Primary key; automatically assigned. Required.	Long - Auto	ID
Last name of this author. Required.	Text(50)	LastName
First name of this author. Optional.	Text(30)	FirstName
Middle name or initial of this author. Optional.	Text(30)	MiddleName
Name suffix, such as "Jr." Optional.	Text(10)	Suffix
Year of birth. Use negative numbers for BC. Optional	Integer	BirthYear
Year of death. Use negative numbers for BC. Optional.	Integer	DeathYear
Miscellaneous comments about this author. Optional.	Text(250)	Comments

ItemAuthor المؤلف،المحرر،وهكذا،بالنسبة لاسم بند معين.وهذا الجدول ينجز علاقة العديد-إلى-العديد many-to-many بين جدول NamedItem وجدول Author.

Description	Type	Field
Primary key. The associated named item. Foreign reference to NamedItem.ID. Required.	Long	ItemID
Primary key. The author associated with the named item. Foreign reference to Author.ID. Required.	Long	AuthorID
Relative order of this author among the authors for this named item. Authors with smaller numbers appear first. Required.	Integer	Sequence
The specific type of contribution given by this author for this named item. Foreign reference to CodeAuthorType.ID. Required.	Long	AuthorType

Keyword كلمات خاصة يمكن تطبيقها على البنود المعينة بالاسم named items لجعل البحث أكثر سهولة.

Description	Type	Field
Primary key; automatically assigned. Required.	Long - Auto	ID
Name of this keyword. Required.	Text(50)	FullName

ItemKeyword يصل keyword بالبند المحدد بالاسم(بالبند المسمى named item) من خلال علاقة عديد-إلى-عديد بين جدول NamedItem وجدول Keyword.

Description	Type	Field
Primary key. The associated named item. Foreign reference to	Long	ItemID

Description	Type	Field
NamedItem.ID. Required.		
Primary key. The keyword to associate with the named item. Foreign reference to Keyword.ID. Required.	Long	KeywordID

Subject عنونة المواضيع Subject headings المستخدمة لتصنيف البنود المسماة(المحددة بالاسم).

Description	Type	Field
Primary key; automatically assigned. Required.	Long - Auto	ID
Name of this subject. Required.	Text(150)	FullName

ItemSubject تصل الموضوع مع البند المسمى من خلال علاقة عديد-إلى-عديد بين جدول NamedItem وجدول Subject.

Description	Type	Field
Primary key. The associated named item. Foreign reference to NamedItem.ID. Required.	Long	ItemID
Primary key. The subject to associate with the named item. Foreign reference to Subject.ID. Required.	Long	SubjectID

جداول المرتبطة بالزبائن Patron-related tables

الجداول في هذا المقطع تعرف سجلات الزبون الحقيقية وعلاقتها مع نسخ البند item copies (عندما يتم تفحص مخرجات مثل هذه النسخ بواسطة الزبون) .

Patron.مستخدم محدد للمكتبة،وعادة للزبائن Patrons امتيازات privileges فيما يخص المخرجات(checkout).

Description	Type	Field
Primary key; automatically assigned. Required.	Long - Auto	ID
Last name of this patron. Required.	Text(30)	LastName
First name of this patron. Required.	Text(30)	FirstName
Date of last checkout, renewal, or return. Optional.	Date	LastActivity
Is this an active patron? 0 for False, 1 for True. Required.	Boolean	Active
Any comments associated with this patron. Optional.	Memo	Comments
Comments that are displayed to administrative users when the patron's record is accessed. Optional.	Memo	AdminMessage
Bar code found on this patron's library card. At this time, only numeric bar codes are supported. Optional.	Text(20)	Barcode
Patron's password, in an encrypted format. Required.	Text(20)	Password
Patron's email address. Optional.	Text(100)	Email
Patron's phone number. Optional.	Text(20)	Phone
Patron's street address. Optional.	Text(50)	Address
Patron's city. Optional.	Text(20)	City
Patron's state abbreviation. Optional.	Text(2)	State
Patron's postal code. Optional.	Text(10)	Postal
The group in which this patron appears. Foreign reference to CodePatronGroup.ID. Optional.	Long	PatronGroup

PatronCopy يدير هذا الجدول نسخ البنود item copies المراجعة حالياً بواسطة الزبون،أو نسخ بند تم مراجعتها سابقاً وقد يتم مراجعتها فيما بعد .

Description	Type	Field
Primary key; automatically assigned. Required.	Long - Auto	ID
The associated patron. Foreign reference to Patron.ID. Required.	Long	Patron
The item copy currently or previously checked out by the patron. Foreign reference to ItemCopy.ID. Required.	Long	ItemCopy
The date when this item copy was initially checked out. Required.	Date	CheckOut
The number of times this item copy has been renewed. Set to 0 when the item copy is first checked out. Required.	Integer	Renewal

Field	Type	Description
DueDate	Date	Current due date for this item copy. Required.
CheckIn	Date	The date when this item copy was returned. Optional.
Returned	Boolean	Has the item copy been returned? 0 for False, 1 for True. Required.
Missing	Boolean	Is the item copy missing and considered lost? 0 for False, 1 for True. Required.
Fine	Currency	Total fine accumulated for this item copy. Defaults to 0.00. An administrator may reduce an accumulated fine. Required.
Paid	Currency	Total amount paid (in fees) for this item copy. Required.
ProcessDate	Date	When an item copy is processed for overdue fines, this field contains the last date for which processing was done. Optional.

PatronPayment.الغرامات Fines، التسديدات(المدفوعات payments)، الطرد dismissals على سجل نسخة الزبون patron copy record.غرامات التأخير Overdue fines غير مسجلة في هذا الجدول،ولكن الغرامات الموضوعة من قبل المدير تبعاً للمسؤوليات بالنسبة للبنود المفقودة تم تسجيلاها هنا .

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required.
PatronCopy	Long	The associated item checked out by the patron. Foreign reference to PatronCopy.ID. Required.
EntryDate	Date	Date and time when this entry was recorded. Required.
EntryType	Text(1)	The type of payment entry. Required. The possible values are: • "P" - The patron made a payment. • "F" - A fine (other than a standard overdue fine) was imposed by an administrator. • "D" - A portion (or all) of the fine was dismissed. • "R" - A refund was given to the patron due to overpayment.
Amount	Currency	The amount associated with this entry. The value is always positive. Required.
Comment	Text(50)	A short comment about this entry. Optional.
UserID	Long	The user who added this payment event. Foreign reference to UserName.ID. Optional.

الجداول المتعلقة بكود التعريف Bar code-related tables

يوجد ثلاث مستويات لتعريف أو إنشاء كود تعريف:(1)الصفيحة التي عليها تطبع شبكة من العناوين،(2)عنوان مفرد single label على صفيحة(صفحة sheet)،(3) البنود المستقلة التي تظهر على كل عنوان.الجداول الثلاث في هذا المقطع تعرف هذه المستويات الثلاث.

BarcodeSheet يصف قالب خاص بصفحة مفردة من عناوين كود التعريف .

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required.
FullName	Text(50)	Name of this sheet template. Required.
UnitType	Text(1)	Units used in the various measurements found in most fields in this record. Required. • I = Inches • C = Centimeters • P = Points • T = Twips
PageWidth	Number	Width of the entire page. Required.
PageHeight	Number	Height of the entire page. Required.
MarginLeft	Number	Left border, up to the edge of the printable label area. Required.
MarginRight	Number	Right border, up to the edge of the printable label area. Required.

Description	Type	Field
Top border, up to the edge of the printable label area. Required.	Number	MarginTop
Bottom border, up to the edge of the printable label area. Required.	Number	MarginBottom
The width of the blank area between label columns. Required.	Number	IntraColumn
The height of the blank area between label rows. Required.	Number	IntraRow
The number of label columns on this template. Required.	Integer	ColumnsCount
The number of label rows on this template. Required.	Integer	RowsCount

BarcodeLabel يصف قالب عنوان مفرد على صفيحة كود تعريف.أي عدد من العناوين يمكن أن تكون على صفيحة مفردة،ولكن كلها لديها نفس الشكل shapeوالهيئة(التنسيق format).

Description	Type	Field
Primary key; automatically assigned. Required.	Long - Auto	ID
Name of this label template. Required.	Text(50)	FullName
The sheet template on which this label template appears. Foreign reference to BarcodeSheet.ID. Required.	Long	BarcodeSheet
Units used in the various measurements found in most fields in this record. Required. • I = Inches • C = Centimeters • P = Points • T = Twips	Text(1)	UnitType

BarcodeLabelItem يصف بند مفرد عندما يوجد على عنوان كود تعريف.تتضمن البنود النصوص الثابتة والمولدة،الأسطر،المستطيلات،و أكواد التعريف المولدة .

Description	Type	Field
Primary key; automatically assigned. Required.	Long - Auto	ID
Identifies the order in which items on the label are printed. Lower numbers are printed first. Required.	Integer	Priority
The label template on which this item appears. Foreign reference to BarcodeLabel.ID. Required.	Long	BarcodeLabel
What type of item does this record represent? Required. • T = Static text • B = Barcode • N = Barcode number • L = Line • R = Rectangle	Text(1)	ItemType
Left edge of the item relative to the left edge of the label. Measured according to the related BarcodeLabel.UnitType field. Required.	Number	PosLeft
Top edge of the item relative to the top edge of the label. Measured according to the related BarcodeLabel.UnitType field. Required.	Number	PosTop
Width of the item, or of the box in which the item is drawn. For lines, this is the x-coordinate of the end point. Measured according to the related BarcodeLabel.UnitType field. Required.	Number	PosWidth
Height of the item, or of the box in which the item is drawn. For lines, this is the y-coordinate of the end point. Measured according to the related BarcodeLabel.UnitType field. Required.	Number	PosHeight
Rotation angle, in degrees, of the box in which the item is drawn. Zero (0) equals no angle, and increasing angles proceed	Integer	Rotation

Field	Type	Description
		clockwise. Ranges from 0 to 359. Only used when ItemType is T, B, N, or R. Optional.
FontName	Text(50)	The name of the font used to write the text. Valid only when ItemType is T or N. Optional.
FontSize	Number	The size of the font used to write the text. Valid only when ItemType is T, B, or N. Optional.
StaticText	Text(100)	The static text to display on the label. Valid only when ItemType is T. Optional.
FontStyle	Text(4)	The style of the font text. May be any combination of the following four codes. • B = Bold • I = Italic • U = Underline • K = Strikeout Leave this field NULL to use the normal style. Valid only when ItemType is T or N. Optional.
Color1	Long	The main color of the text, bar code, or line. When printing a rectangle, this is the border color. If NULL, black is used. A standard Windows 32-bit RGB color value. Optional.
Color2	Long	The fill color when printing a rectangle. If NULL, white is used. A standard Windows 32-bit RGB color value. Optional.
Alignment	Integer	The alignment of the text within the bounding box. Valid only when ItemType is T, B, or N. • 1 = Align in top-left corner of box • 2 = Align in top-center area of box • 4 = Align in top-right corner of box • 16 = Align in middle-left area of box • 32 = Align in middle-center area of box • 64 = Align in middle-right area of box • 256 = Align in bottom-left corner of box • 512 = Align in bottom-center area of box • 1024 = Align in bottom-right corner of box
PadDigits	Integer	The number of digits in which to pad the bar code number. Set to zero (0) to ignore padding. Ranges from 0 to 20. If the bar code length is less than the specified number of digits, it is padded on the left with zeros. Only applies to ItemTypes of B and N.

جداول أخرى متنوعة (شتى) Other miscellaneous tables

يوفر جدولين إضافيين دعم للميزات التي لم يتم معالجتها من خلال الجداول الأخرى.

Holiday عندما مراجعة بند ما لزبون،فإن التاريخ الراجع يجب أن لايتضمن العطل(أو أي يوم تكون فيه المكتبة مغلقة)،حيث أن الزبون من المحتمل أن لاتكون لديه طريقة لإعادة الكتاب في مثل هذا اليوم.يحدد(يعرف) هذا الجدول سابقاً ويرجع العطل recurring holidays.

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required.
FullName	Text(50)	Name of this holiday. Not necessarily unique. Required.
EntryType	Text(1)	The type of entry. Required. From the following list. • A = Annual (as in “every December 25”) • E = Weekly (as in “every Sunday”) • O = One-time (as in “2/16/2004 is President’s Day”)
EntryDetail	Text(10)	Entry-type-specific detail. Required. Differs for each entry type.

Field	Type	Description
		Entry Type Detail Value A Month and Day in “mm/dd” format E Single digit: 1=Sunday through 7=Saturday O Date in “yyyy/mm/dd” format

SystemValue يخزن هذا الجدول إعدادات واسعة ونهائية شتى miscellaneous enterprise-wide settingsوالتي تطبق على كل شبكة حاسوبworkstation . يتم تخزين الإعدادات الخاصة بشبكة حاسوب محلية Local workstation-specific settings على كل جهاز،وليس في قاعدة البيانات.

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required.
ValueName	Text(50)	Name of this value. Required.
ValueData	Text(100)	Information associated with this entry. Optional.

يتم تعريف قيم النظام التالية في الوقت الحالي.واسم الكود يظهر في حقل ValueName.والقيمة الموافقة تظهر في حقل ValueData.

BarcodeCode39

هل كود التعريف مخصص في تنسيق "الكود39"”code 39“أو"كود3من 9"”code 3 of 9“؟إذا كان كذلك،سيتم وضع نجمة قبل وبعد رقم كود التعريف bar code number قبل أن يتم طباعته على عنوان ما.استخدم القيمة0 لخطأ Falseوأي قيمة غير الصفر لصواب(-1 مفضلة).ويتم افتراض خطأ في حالة فقدان القيمة أو بدون قيمة missing or NULL.

BarcodeFont

اسم الخط المستخدم لطباعة أكواد التعريف.هذا الخط يجب أن يكون مثبت على أي شبكة حاسوب تعرض أو تطبع أكواد التعريف.وليس هناك حاجة لعمل مسح على أكواد التعريف.

DatabaseVersion

أي إصدار هيكلي لقاعدة البيانات قيد الاستخدام حالياً؟يتم وضعه حالياً إلى"1"،ويتم حفظه من أجل التحسينات المستقبلية.

DefaultLocation

قيمة حقل CodeLocation.ID من أجل الموقع locationالذي يتم وضعه كافتراضي .

FineGrace(مهلة الغرامة)

عدد الأيام التي يمكن لبند أن يتأخر ضمنها بند ما دون أن تتسبب في غرامة without incurring a fine.

NextBarcodeItem

قيمة البداية التالية لتستخدم عند طباعة أكواد تعريف بندitem bar codes .

NextBarcodeMisc

قيمة البداية التالية لتستخدم عند طباعة أكواد التعريف المختلفة(الشتى) miscellaneous bar codes.

NextBarcodePatron

قيمة البداية التالية لتستخدم عند طباعة أكواد تعريف العميل (الزبون)patron bar codes).

PatronCheckOut

تشير فيما إذا بإمكان الزبائن استخراج بنود دون الحاجة لتسجيل الدخول كمستخدمين إداريين.استخدم القيمة 0(صفر)لتشير إلى عدم وجود امتيازات استخراج وقيمة غير الصفر للسماح لزبون من المراجعة(-1 مفضلة).فإذا ما فقدت هذه القيمة أو كانت فارغة،فلن يتم السماح للزبائن من استخراج بنود دون مساعدة المدير.

SearchLimit

تشير إلى العدد الأكبر من النتائج المعادة في أي عملية بحث.فإذا مافقدت هذه القيمة أو كانت غير صحيحة،يتم استخدام الافتراضي 250.والمجال المسموح الشامل inclusive هو بين 25و5000.

TicketHeading

يعرض النص الذي يجب طباعته عند أعلى بطاقات مخرجات الدفع.كل الأسطر تتوسط البطاقة.بما فيه حرف الشريط العمودي(|) لتقسيم النص في عدة أسطر .

TicketFooting

يعرض النص الذي سيتم طباعته أسفل بطاقات مخرجات الدفع.كل الأسطر تتوسط البطاقة.بما فيها حرف الشريط العمودي(|) لتقسيم النص في عدة أسطر.

UseLC

يشير فيما إذا تم استدعاء أرقام تصنيف كتاب بواسطة Deweyأو مكتبة الكونغرس(في أمريكا) (Library of Congress (LC.استخدم القيمة 0(صفر)لتشير إلى Dewey،أو قيمة غير الصفر لـ LC(-1 مفضلة).فإذا ما فقدت هذه القيمة أو كانت فارغة فالافتراضي هو Dewey.

إنشاء قاعدة البيانات Creating the Database

إضافة قاعدة بيانات إلى سكول سرفر هي عملية سهلة بقدر سهولة توثيقها،في الحقيقة،تتطلب كتابة أقل.وعبارات CREATE TABLE بسيطة جداً،وتبدو جميعها متشابهة إلى حدٍ بعيد.وسأريك فقط بعضاً منها هنا.(يحتوي ملف*Database CreationScript.sql* على المحتوى الكامل للصيغ وقد قمت بوضعه في برامج الفصل الرابع).

التعليمات المجدولة هنا هي من أجل إدارة منصة سكول سرفر2005 السريعةSQL Server 2005 Management Studio Express. بإمكانك إتمام كل هذه المهام باستخدام إدارة منصة سكول سرفر2005،أو حتى أدوات أسطر الأمر command-line toolsالموفرة بواسطة سكول سرفر،ولكن قد تتفاوت التفاصيل بالنسبة لكل خطوة.تعمل نفس عبارات CREATE TABLE مع أي أداة تختارها.إذا لم تكن قد عملت مثل هذا سابقاً. اعمل على تثبيت

سكول سرفر 2005 بطبعته السريعةSQL Server 2005 Express Edition (أو أي إصدار من قاعدة البيانات سيكون بإمكانك استخدامه).إن إدارة منصة سكول سرفر2005 السريعة منتج مفصول عن سكول سرفر نفسه،لذلك يجب أن تعمل على تثبيتها أيضاً.

معظم الجداول في مشروع المكتبة هي جداول بيانات بسيطة ذات مفتاح رئيسي مفرد.وكودها بسيط جداً.فجدول المؤلف Authorهو مثال جيد.

```
CREATE TABLE Author
(
ID bigint IDENTITY PRIMARY KEY,
LastName varchar(50) NOT NULL,
FirstName varchar(30) NULL,
MiddleName varchar(30) NULL,
Suffix varchar(10) NULL,
BirthYear smallint NULL,
DeathYear smallint NULL,
Comments varchar(250) NULL
);
```

الحقول المضمنة في كل عبارة CREATE TABLE تظهر كقائمة محددة بفاصلةcomma-delimited list ،وجميعها(الحقول) مغلفة في أقواس.وكل حقل يتضمن إما خيار بدون قيمة NULLأو ليس بدون قيمة NOT NULL والذي يشير فيما إذا سيتم استخدام القيم بدون قيمة NULLفي ذلك الحقل أم لا.

خيار المفتاح الرئيسي PRIMARY KEY بشكل آلي يخصص "ليس بدون قيمةNOT NULL "

بعض العبارات تنشئ جداول تربط جدولين آخرين في علاقة عديد-إلى-عديد.وأحد أمثلتها جدول GroupActivity،الذي يصل جدول GroupNameمع جدول Activity.

```
CREATE TABLE GroupActivity
(
GroupID bigint NOT NULL,
ActivityID bigint NOT NULL,
PRIMARY KEY (GroupID, ActivityID)
);
```

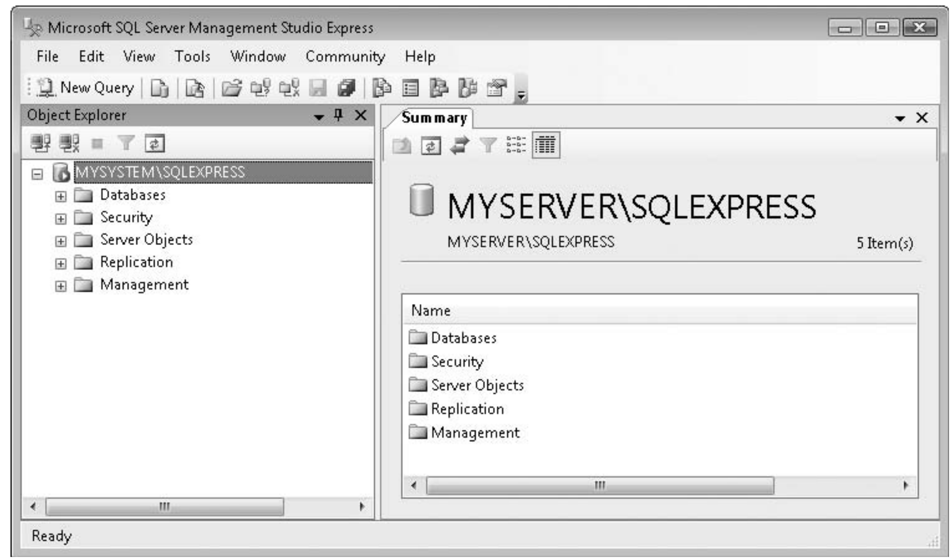
لجدول المؤلف Authorمفتاح رئيسي مفرد،لذلك خيار المفتاح الرئيسي يمكن أن يتم إلحاقه مباشرةً لحقل المعرف IDالتابع له. بما أن جدول GroupActivity لديه حقليّ مفتاح رئيسي(وهو معروف في قاعدة البيانات العلائقية)،خيار المفتاح الرئيسي PRIMARY KEY يتم تخصيصه كمدخلة خاصة به تماماً،مع حقول المفاتيح المخصصة ضمن القوسين والمفصولة فيما بينها بفاصلة PRIMARY KEY (GroupID, ActivityID).

سابقاً في هذا الفصل ،بينت لك كيف تستطيع تأسيس علاقة لحقل في جدول أخر باستخدام قيود مرجعية(علائقيةREFERENCES constraint)كجزء من عبارة CREATE TABLE.بإمكانك أيضاً تأسيسها بعد أن يتم وضع الجداول في مكانها.(كالذي عملته في ملف صيغة سكول).إليك العبارة التي تؤسس اتصال بين جدول GroupActivityوجدول GroupName.

```
ALTER TABLE GroupActivity
ADD FOREIGN KEY (GroupID)
REFERENCES GroupName (ID);
```

حيث أنني قمت بكتابة صيغ سكول الكاملة entire SQL script لك،فإنني أطلب منك فقط معالجتها مباشرةً باستخدام إدارة منصة ميكروسوفت سكول سرفر2005 السريعةMicrosoft SQL Server 2005 Management Studio Express .(إذا كنت ستستخدم الإصدار الكامل من سكول سرفر أو بعض الأدوات الإدارية management tool الأخرى،فإن الصيغ الموفرة ستبقى تعمل ،على الرغم من أن تعليمات خطوة-بخطوة ستختلف).قبل إضافة الجداول tables،نحتاج إلى إنشاء قاعدة بيانات مخصصة لمشروع المكتبةLibrary Project .شغل منصة إدارة ميكروسوفت سكول سرفر2005 السريعة (شاهد الشكل التالي)لإضافة قاعدة بيانات جديدة new database لمشروع المكتبة،انقر يمين على مجلد "قاعدة البيانات *Database* " في مستكشف الكائن،واختر "قاعدة بيانات جديدةNew Database from " من القائمة المختصرة.على نموذج قاعدة البيانات الجديد الذي يظهر،أدخل **Library**في حقل اسم قاعدة Database Name،ومن ثم انقر موافق OK.

قاعدة بيانات المكتبة مجرد غلاف قاعدة بيانات،فهي لا تحتوي أي جدول أو أي بيانات حتى الآن.لنستخدم ملف *Database Creation Script.sql* لإنتاج الجداول والبيانات الأولية.في إدارة المنصة السريعة،اختر ملف File>> فتح Open>> أمر قائمة ملف File menu command،وأوجد ملف *Database Creation Script.sql*.(من الممكن أن يطلب منك تسجيل الدخول لسكول سرفر مرةً أخرى) يفتح هذا الملف مساحات لمحتوياته في لوحة جديدة ضمن إدارة المنصة السريعة Management Studio Express.



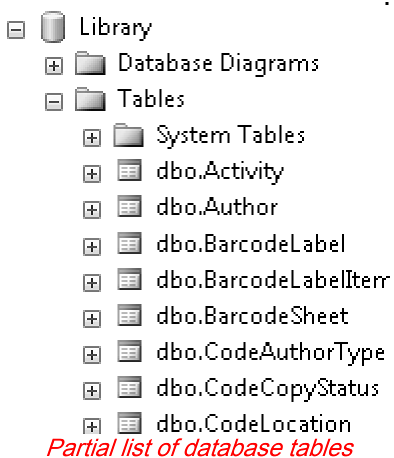
SQL Server 2005 Management Studio Express main form

كل ما بقي عليك عمله هو معالجة الصيغةscript. في منطقة "شريط الأدوات toolbar"،تأكد من أن " Library" هي قاعدة البيانات المختارة(شاهد الشكل التالي).ومن ثم انقر على زر تنفيذ Executeمن شريط الأدوات،أو اضغط مفتاح F5.وهي صيغة صغيرة مع القليل من العوامل not a lot going on (على الأقل من وجهة نظر سكول سرفر)،لذلك سيتم الانتهاء منها في عدة ثواني.



هذا كل شيء،أغلق لوحة الصيغةscript panel. ومن ثم، ارجع لمستكشف الكائنObject Explorer، وانقر يمين على مجلد قاعدة بيانات المكتبة *Library* واختر تحديث Refreshمن القائمة. فإذا ما وسعت بعدئذٍ تفرع قاعدة بيانات المكتبة وجداولها المتفرعة،سترى كل الجداول المنشئة بواسطة الصيغة (شاهد الشكل التالي).

والآن بعد إتمام قاعدة البيانات،حان الوقت لتبدأ البرمجة .



Partial list of database tables

سأضمن لك ملف*Database Creation Script.sql* كاملا :

```
----- */ACME Library Project
     * Database creation script.
     * Start-to-Finish Visual Basic 2005
     * Copyright (c) 2006 by Tim Patrick
/*

CREATE TABLE Activity
)
  ID       bigint      PRIMARY KEY،
  FullName   varchar(50)   NOT NULL
؛(
GO
```

```
INSERT INTO Activity (ID, FullName) VALUES (1, 'Manage authors and names؛('
INSERT INTO Activity (ID, FullName) VALUES (2, 'Manage author and name types؛('
INSERT INTO Activity (ID, FullName) VALUES (3, 'Manage copy status codes؛('
INSERT INTO Activity (ID, FullName) VALUES (4, 'Manage media types؛('
INSERT INTO Activity (ID, FullName) VALUES (5, 'Manage series؛('
INSERT INTO Activity (ID, FullName) VALUES (6, 'Manage security groups؛('
INSERT INTO Activity (ID, FullName) VALUES (7, 'Manage library materials؛('
INSERT INTO Activity (ID, FullName) VALUES (8, 'Manage patrons؛('
INSERT INTO Activity (ID, FullName) VALUES (9, 'Manage publishers؛('
INSERT INTO Activity (ID, FullName) VALUES (10, 'Manage system values؛('
INSERT INTO Activity (ID, FullName) VALUES (11, 'Manage administrative users؛('
INSERT INTO Activity (ID, FullName) VALUES (12, 'Process and accept fees؛('
INSERT INTO Activity (ID, FullName) VALUES (13, 'Manage locations؛('
INSERT INTO Activity (ID, FullName) VALUES (14, 'Check out library items؛('
INSERT INTO Activity (ID, FullName) VALUES (15, 'Check in library items؛('
INSERT INTO Activity (ID, FullName) VALUES (16, 'Access administrative features؛('
INSERT INTO Activity (ID, FullName) VALUES (17, 'Perform daily processing؛('
INSERT INTO Activity (ID, FullName) VALUES (18, 'Run system reports؛('
INSERT INTO Activity (ID, FullName) VALUES (19, 'Access patrons without patron password؛('
INSERT INTO Activity (ID, FullName) VALUES (20, 'Manage barcodes؛('
INSERT INTO Activity (ID, FullName) VALUES (21, 'Manage holidays؛('
INSERT INTO Activity (ID, FullName) VALUES (22, 'Manage patron groups؛('
INSERT INTO Activity (ID, FullName) VALUES (23, 'View administrative patron messages؛('
GO

CREATE TABLE Author
)
   ID          bigint        IDENTITY PRIMARY KEY،
   LastName    varchar(50)   NOT NULL،
   FirstName   varchar(30)   NULL،
   MiddleName  varchar(30)   NULL،
   Suffix      varchar(10)   NULL،
   BirthYear   smallint      NULL،
   DeathYear   smallint      NULL،
   Comments    varchar(250)  NULL
؛(
GO

CREATE TABLE BarcodeLabel
)
   ID            bigint       IDENTITY PRIMARY KEY،
   FullName      varchar(50)  NOT NULL،
   BarcodeSheet  bigint       NOT NULL،
   UnitType      varchar(1)   NOT NULL
؛(
GO

CREATE TABLE BarcodeLabelItem
)
   ID            bigint       IDENTITY PRIMARY KEY،
```

```
   Priority      smallint      NOT NULL،
   BarcodeLabel  bigint        NOT NULL،
   ItemType      varchar(1)    NOT NULL،
   PosLeft       decimal(10,4)  NOT NULL،
   PosTop        decimal(10,4)  NOT NULL،
   PosWidth      decimal(10,4)  NOT NULL،
   PosHeight     decimal(10,4)  NOT NULL،
   Rotation      smallint      NULL،
   FontName      varchar(50)   NULL،
   FontSize      decimal(10,4)  NULL،
   StaticText    varchar(100)   NULL،
   FontStyle     varchar(4)    NULL،
   Color1        bigint        NULL،
   Color2        bigint        NULL،
   Alignment     smallint      NULL،
   PadDigits     smallint      NULL
؛(
GO

CREATE TABLE BarcodeSheet
)
   ID            bigint        IDENTITY PRIMARY KEY،
   FullName      varchar(50)   NOT NULL،
   UnitType      varchar(1)    NOT NULL،
   PageWidth     decimal(10,4)  NOT NULL،
   PageHeight    decimal(10,4)  NOT NULL،
   MarginLeft    decimal(10,4)  NOT NULL،
   MarginRight   decimal(10,4)  NOT NULL،
   MarginTop     decimal(10,4)  NOT NULL،
   MarginBottom  decimal(10,4)  NOT NULL،
   IntraColumn   decimal(10,4)  NOT NULL،
   IntraRow      decimal(10,4)  NOT NULL،
   ColumnsCount  smallint      NOT NULL،
   RowsCount     smallint      NOT NULL
؛(
GO

CREATE TABLE CodeAuthorType
)
   ID        bigint       IDENTITY PRIMARY KEY،
   FullName  varchar(50)  NOT NULL
؛(
GO

CREATE TABLE CodeCopyStatus
)
   ID        bigint       IDENTITY PRIMARY KEY،
   FullName  varchar(50)  NOT NULL
؛(
GO

CREATE TABLE CodeLocation
```

```
)
  ID            bigint        IDENTITY PRIMARY KEY،
  FullName        varchar(50)    NOT NULL،
  LastProcessing  datetime       NULL
؛(
GO

CREATE TABLE CodeMediaType
)
  ID            bigint       IDENTITY PRIMARY KEY،
  FullName       varchar(50)   NOT NULL،
  CheckoutDays   smallint      NOT NULL،
  RenewDays      smallint      NOT NULL،
  RenewTimes     smallint      NOT NULL،
  DailyFine      money         NOT NULL
؛(
GO

CREATE TABLE CodePatronGroup
)
  ID        bigint       IDENTITY PRIMARY KEY،
  FullName   varchar(50)   NOT NULL
؛(
GO

CREATE TABLE CodeSeries
)
  ID        bigint       IDENTITY PRIMARY KEY،
  FullName   varchar(50)   NOT NULL
؛(
GO

CREATE TABLE GroupActivity
)
  GroupID      bigint   NOT NULL،
  ActivityID   bigint   NOT NULL،
  PRIMARY KEY (GroupID, ActivityID(
؛(
GO

CREATE TABLE GroupName
)
  ID        bigint       IDENTITY PRIMARY KEY،
  FullName   varchar(50)   NOT NULL
؛(
GO

CREATE TABLE Holiday
)
  ID           bigint       IDENTITY PRIMARY KEY،
  FullName      varchar(50)   NOT NULL،
  EntryType     varchar(1)    NOT NULL،
```

```
  EntryDetail   varchar(10)   NOT NULL
؛(
GO

CREATE TABLE ItemAuthor
)
  ItemID      bigint     NOT NULL،
  AuthorID    bigint     NOT NULL،
  Sequence     smallint   NOT NULL،
  AuthorType   bigint     NOT NULL،
  PRIMARY KEY (ItemID, AuthorID(
؛(
GO

CREATE TABLE ItemCopy
)
  ID         bigint       IDENTITY PRIMARY KEY،
  ItemID      bigint       NOT NULL،
  CopyNumber   smallint      NOT NULL،
  Description  varchar(max)  NULL،
  Available    bit          NOT NULL،
  Missing      bit          NOT NULL،
  Reference    bit          NOT NULL،
  Condition    varchar(30)   NULL،
  Acquired     datetime      NULL،
  Cost        money        NULL،
  Status       bigint       NOT NULL،
  Barcode      varchar(20)   NULL،
  Location     bigint       NULL
؛(
GO

CREATE TABLE ItemKeyword
)
  ItemID      bigint   NOT NULL،
  KeywordID   bigint   NOT NULL،
  PRIMARY KEY (ItemID, KeywordID(
؛(
GO

CREATE TABLE ItemSubject
)
  ItemID      bigint   NOT NULL،
  SubjectID   bigint   NOT NULL،
  PRIMARY KEY (ItemID, SubjectID(
؛(
GO

CREATE TABLE Keyword
)
  ID        bigint       IDENTITY PRIMARY KEY،
  FullName   varchar(50)   NOT NULL
```

```
;(
GO

CREATE TABLE NamedItem
)
  ID          bigint       IDENTITY PRIMARY KEY،
  Title       varchar(150)   NOT NULL،
  Subtitle    varchar(150)   NULL،
  Description  varchar(max)   NULL،
  Edition     varchar(10)    NULL،
  Publisher   bigint        NULL،
  Dewey       varchar(20)    NULL،
  LC          varchar(25)    NULL،
  ISxN        varchar(20)    NULL،
  LCCN        varchar(12)    NULL،
  Copyright   smallint      NULL،
  Series      bigint        NULL،
  MediaType   bigint        NOT NULL،
  OutOfPrint  bit           NOT NULL
;(
GO

CREATE TABLE Patron
)
  ID          bigint       IDENTITY PRIMARY KEY،
  LastName      varchar(30)    NOT NULL،
  FirstName     varchar(30)    NOT NULL،
  LastActivity  datetime      NULL،
  Active        bit           NOT NULL،
  Comments      varchar(max)   NULL،
  AdminMessage  varchar(max)   NULL،
  Barcode       varchar(20)    NULL،
  Password      varchar(20)    NOT NULL،
  Email         varchar(100)   NULL،
  Phone         varchar(20)    NULL،
  Address       varchar(50)    NULL،
  City          varchar(20)    NULL،
  State         varchar(2)     NULL،
  Postal        varchar(10)    NULL،
  PatronGroup   bigint  NULL
;(
GO

CREATE TABLE PatronCopy
)
  ID          bigint    IDENTITY PRIMARY KEY،
  Patron      bigint    NOT NULL،
  ItemCopy    bigint    NOT NULL،
  CheckOut    datetime   NOT NULL،
  Renewal     smallint   NOT NULL،
  DueDate     datetime   NOT NULL،
  CheckIn     datetime   NULL،
```

```
  Returned     bit       NOT NULL،
  Missing      bit       NOT NULL،
  Fine         money     NOT NULL،
  Paid         money     NOT NULL،
  ProcessDate  datetime  NULL
؛(
GO

CREATE TABLE PatronPayment
)
  ID          bigint        IDENTITY PRIMARY KEY،
  PatronCopy  bigint        NOT NULL،
  EntryDate   datetime      NOT NULL،
  EntryType   varchar(1)    NOT NULL،
  Amount      money         NOT NULL،
  Comment     varchar(50)   NULL،
  UserID      bigint        NULL
؛(
GO

CREATE TABLE Publisher
)
  ID        bigint        IDENTITY PRIMARY KEY،
  FullName  varchar(100)  NOT NULL،
  WebSite   varchar(255)  NULL
؛(
GO

CREATE TABLE Subject
)
  ID        bigint        IDENTITY PRIMARY KEY،
  FullName  varchar(150)  NOT NULL
؛(
GO

CREATE TABLE SystemValue
)
  ID         bigint        IDENTITY PRIMARY KEY،
  ValueName  varchar(50)   NOT NULL،
  ValueData  varchar(100)  NULL
؛(
GO

INSERT INTO SystemValue (ValueName, ValueData) VALUES ('BarcodeCode39', '0؛('
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('BarcodeFont', NULL؛(
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('DatabaseVersion', '1؛('
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('DefaultLocation', NULL؛(
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('FineGrace', '3؛('
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('Licensee', NULL؛(
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('LicenseCode', NULL؛(
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('NextBarcodeItem', '2000001؛('
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('NextBarcodeMisc', '1000001؛('
```

```
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('NextBarcodePatron', '1؛('
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('PatronCheckOut', '-1؛('
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('SearchLimit', '250؛('
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('TicketHeading', 'Library System؛('
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('TicketFooting', 'Thank you؛('.
INSERT INTO SystemValue (ValueName, ValueData) VALUES ('UseLC', '0؛('
GO

CREATE TABLE UserName
)
   ID        bigint       IDENTITY PRIMARY KEY،
   FullName   varchar(50)   NOT NULL،
   LoginID    varchar(20)   NOT NULL،
   Password   varchar(20)   NULL،
   Active     bit          NOT NULL،
   GroupID    bigint       NOT NULL
؛(
GO

ALTER TABLE GroupActivity    ADD FOREIGN KEY (GroupID)      REFERENCES GroupName      (ID؛(
ALTER TABLE GroupActivity    ADD FOREIGN KEY (ActivityID)   REFERENCES Activity       (ID؛(
ALTER TABLE UserName         ADD FOREIGN KEY (GroupID)      REFERENCES GroupName      (ID؛(
ALTER TABLE NamedItem        ADD FOREIGN KEY (Publisher)    REFERENCES Publisher      (ID؛(
ALTER TABLE NamedItem        ADD FOREIGN KEY (Series)       REFERENCES CodeSeries     (ID؛(
ALTER TABLE NamedItem        ADD FOREIGN KEY (MediaType)    REFERENCES CodeMediaType   (ID؛(
ALTER TABLE ItemCopy         ADD FOREIGN KEY (ItemID)       REFERENCES NamedItem      (ID؛(
ALTER TABLE ItemCopy         ADD FOREIGN KEY (Status)       REFERENCES CodeCopyStatus  (ID؛(
ALTER TABLE ItemCopy         ADD FOREIGN KEY (Location)     REFERENCES CodeLocation   (ID؛(
ALTER TABLE ItemAuthor       ADD FOREIGN KEY (ItemID)       REFERENCES NamedItem      (ID؛(
ALTER TABLE ItemAuthor       ADD FOREIGN KEY (AuthorID)     REFERENCES Author         (ID؛(
ALTER TABLE ItemAuthor       ADD FOREIGN KEY (AuthorType)   REFERENCES CodeAuthorType  (ID؛(
ALTER TABLE ItemKeyword      ADD FOREIGN KEY (ItemID)       REFERENCES NamedItem      (ID؛(
ALTER TABLE ItemKeyword      ADD FOREIGN KEY (KeywordID)    REFERENCES Keyword        (ID؛(
ALTER TABLE ItemSubject      ADD FOREIGN KEY (ItemID)       REFERENCES NamedItem      (ID؛(
ALTER TABLE ItemSubject      ADD FOREIGN KEY (SubjectID)    REFERENCES Subject        (ID؛(
ALTER TABLE Patron           ADD FOREIGN KEY (PatronGroup)  REFERENCES CodePatronGroup (ID؛(
ALTER TABLE PatronCopy       ADD FOREIGN KEY (Patron)       REFERENCES Patron         (ID؛(
ALTER TABLE PatronCopy       ADD FOREIGN KEY (ItemCopy)     REFERENCES ItemCopy       (ID؛(
ALTER TABLE PatronPayment    ADD FOREIGN KEY (PatronCopy)   REFERENCES PatronCopy     (ID؛(
ALTER TABLE PatronPayment    ADD FOREIGN KEY (UserID)       REFERENCES UserName       (ID؛(
ALTER TABLE BarcodeLabel     ADD FOREIGN KEY (BarcodeSheet) REFERENCES BarcodeSheet   (ID؛(
ALTER TABLE BarcodeLabelItem ADD FOREIGN KEY (BarcodeLabel) REFERENCES BarcodeLabel   (ID؛(
GO

 ----- */Create the basic security account/* .
INSERT INTO GroupName (FullName(
  VALUES ('Administrators؛('

INSERT INTO GroupActivity (GroupID, ActivityID(
  SELECT 1, ID FROM Activity؛
```

```
INSERT INTO UserName (FullName, LoginID, Active, GroupID(
  VALUES ('Administrator', 'admin', 1, 1؛(
GO
```

المجمعاتAssembly

مجمعات الدوت نت هي مجرد ملفات EXEو DLL(مكتبة الربط الديناميكي dynamic link library).بدون الحاجة لتفعيلها من قبلك،فهي تحتل مساحة تجميعية من القرص.وبما أنها لا تعمل أي شيء أخر،لذلك لنتفحص ماهي هذه المجمعات وماذا تحتوي .

ما هو المجمع What Is an Assembly

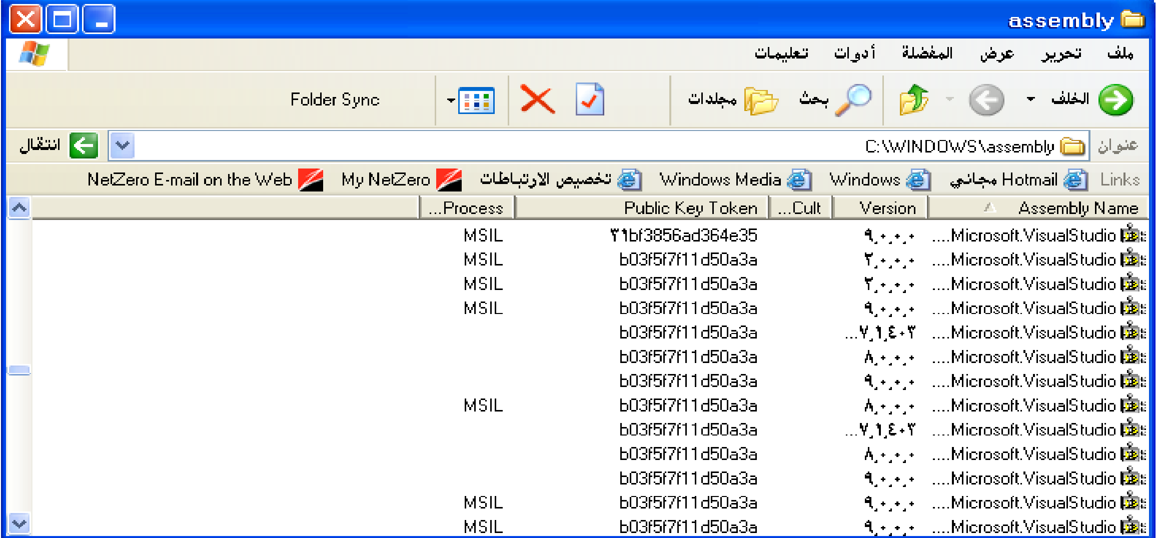
المجمع هو "وحدة نشرunit of deployment "والتي وفي معظم الحالات هي ملف فقط.فالمجمع مستودعrepository معد للترجمة من قبل كود تطبيق دوت نت.فأي كود تكتبه سيتم تخزينه في النهاية في ملف EXE(إذا كان تطبيق) أو ملف DLL(من أجل مكتبات الكود code librariesأو امتدادات تطبيق).كل شيء تحتاج معرفته حول الدوت نت لتحميل وتشغيل تطبيقك يتم تخزينه في مجمع assembly.

المجمعات هي إما خاصة privateأو عامةpublic. يتم تصميم المجمعات الخاصة *Private assemblies* للاستخدام في تطبيق مفرد فقط.فإذا لم يكن هناك أي DLLs،فمجمع EXEهو التطبيق. تظهر المجمعات الخاصة في دليل خاص بها، دليل التنصيب *installation directory* للتطبيق أو المكتبة.

بإمكانك تشغيل مجمعين خاصين مختلفين في نفس الوقت،ولن يتعارضbother كل منهما مع الآخر.وهذا صحيح حتى ولو كان كل مجمع يستخدم نفس تركيب فضاء الأسماء وأسماء الفئة لعناصره المكودة(المشفرة).فإذا مجمعا تطبيق ،كل منهما ينفذ فئة مسماة WindowsApplication1.Class1،فلن يتداخل interfereكل منهما مع الآخر عند التشغيل،فهما خاصان privateوخاص private يعني خاص private .

يتم تصميم المجمعات العامة Public assembliesللاستخدام المتشارك بين تطبيقات الدوت نت المتعددةmultiple .NET applications. تختلف المجمعات العامة Public assemblies عن المجمعات الخاصة private assemblies في طريقتين رئيسيتين:

- المجمعات العامة دائماً لديها اسم قوي*strong name*،توقيع رقمي مشفر encrypted digital signature متعلق بمجمع ما لضمان أنه يأتي من البائع المسمى named vendor أو المصدرsource . (يمكن للمجمعات الخاصة أن تتضمن اسم قوي،ولكن هذا ليس بالمستوجب).يتم بناء الاسم القوي من اسم المجمع،رقم النسخة(الإصدار)، معلومات ثقافية،الكلمة أو"المفتاح عامpublic key "،والتوقيع الرقميdigital signature المولد من ملف مجمع يحوي الكشف manifest.يتضمن إطار عمل الدوت نت أدوات توليد اسم قوي (*sn.exe*)والذي يساعد في هذه المعالجة،وتتضمن فيجوال أستوديو خيارات تتيح لك من إضافة توقيع رقمي خلال عملية الترجمةcompilation process . (وهي ليست تبويب معين في صفحات خاصيات المشروع project's properties).الاسم القوي لمجمع ما يجب أن(ومن الأفضل أن يكون) مفرد ومميز،إذا ما تشارك مجمعان اسم قوي مشترك،فإنهما نسخ لنفس المجمع.
- يتم تخزين المجمعات العامة في الذاكرة الانتقالية للمجمع الشامل Global Assembly Cache (GAC).على الرغم من أنك تستطيع وضع نسخة من مكوناتك المتشاركة في دليل تنصيب تطبيقكapplication's install directory، فستكون صحيحة التشارك فقط عندما تصل لدليل GAC.يقيم GAC في دليل مسمى مجمع *assembly* ضمن دليل ويندوز للكمبيوترcomputer's Windows directory (على نظامي،فهو في *c:\windows\assembly*)حالما يكون لمجمع الدوت نت اسم قوي مطبق،بإمكانك إضافته إلى GACبواسطة إما سحب الملف لضمن دليل المجمع *assembly* أو استخدام أداة الذاكرة الانتقالية للمجمع الشامل(Global Assembly Cache Tool (*gacutil.exe*. ولا تقلق بالنسبة لملفك الوحيد إذا لم يتجاوب مع الملفات المثبتة installed files.في النسخة المثبتة حديثاً للدوت نتNET. ،يمكن أن تجد تقريباً 400ملف موجودة مسبقاً في دليلGAC ،من ضمنها كل DLLsالخاصة بمكتبة فئة إطار العمل(Framework Class Libraries (FCLs. تتيح لك الدوت نتNET. من تثبيت عدة إصدارات multiple versions من مجمع ما assemblyعلى النظام واستخدامهم في نفس الوقت(مثل هذه المعالجة تدعى التوافقية *versioning*).وهذه تقدم (EXE)والمكتبات(libraries (DLL للتطبيق ،ومن أجل المجمعات الخاصة والمجمعات المتشاركة في GAC.افتح مجلد مجمع GAC's *assembly*،وضع مستكشف المجلدات Explorer folder إلى "عرض التفاصيلDetails "،ومن ثم رتب sortبواسطة اسم المجمعAssembly Name. إذا حركة شريط التمرير للأسفل،سترى نفس الملفات المبينة عدة مرات.يبين الشكل التالي جزء من الذاكرة الانتقاليةcache. يتم جدولة نسختان من"Microsoft.VisualStudio.Windows.Forms" (من ملف*Microsoft. VisualStudio.Windows.Forms.dll*) واحدة مع رقم النسخة 2.0وأخرى مع رقم النسخة 9.0. على الرغم من أنه يوجد عادة علاقة واحد –إلى-واحد بين الملفات والمجمعات،فيمكن أن توجد حالات عندما يتم تركيب مجمع من عدة ملفاتmultiple files. على سبيل المثال،يمكن لتطبيق أن يتضمن ملف صور(رسوميgraphics) خارجي في عرض المجمع assembly view الخاص به.يراقب الدوت نت هذه الملفات بحرص شديد.فإذا تم تعديل modified أي من هذه الملفات،حذفتdeleted،أو من ناحية أخرى تشوهتmaimed، فسوف تعلم بها.فيما يخص الشرح هنا،فإن باقي الفصل سيأخذ بعين الاعتبار المجمعات ذات الملف المفردsingle-file assemblies .



The GAC has this duplication under control

ماذا يوجد داخل المجمع What's Inside an Assembly?

ملف EXEأو DLLللمجمع هو الملف PE "تنفيذي قابل للنقل Portable Execution " القياسي ، نفس تنسيق الملف المستخدم من أجل الملفات القابلة للتنفيذ في غير الدوت نت ومكتبات الكود(مثل الكثير من أي ملف ويندوز EXE أوDLL).ما يجعل ملفات الدوت نت التنفيذية والقابلة للنقلNET PE files. مختلفة هي كل

الأجزاء المحشوة الإضافية extra stuff الموجودة داخله.بشكل عام الكلمة مجمع تشير إلى جمع الأجزاء المتنوعة مع بعضها ضمن وحدة مفردة.في مجمع ما للدوت نت،هذه "الأجزاء المتنوعة"مصممة بشكل خاص للاستخدام مع الدوت نت .يحتوي ملف دوت نت القابل للنقل.NET PE على ثلاث أجزاء رئيسية:

معنون "تنفيذي قابل للنقل" A PE header

مطلوب لكل ملفات PE،هذا المقطع يحدد(يعرف) المواقع بالنسبة للمقاطع الأخرى من الملف.

مقطع كود لغة ميكروسوفت الوسيطة The MSIL code section

يتم تخزين الكود الفعلي المصاحب المجمع ككود لغة ميكروسوفت الوسيطة شبه المترجم semicompiled Microsoft Intermediate Language (MSIL).لسوء الحظ،شريحة chipإنتل Intelأو AMD في كمبيوترك بكل وضوح بلهاء كثيراً لمعالجة كود لغة ميكروسوفت الوسيطة بشكل مباشر،لذلك يتضمن إطار الدوت نت مترجم فوري (JIT)compiler *just-in-time*بإمكانه تحويل لغة ميكروسوفت الوسيطة MSILإلى كود x86الأصلي فوراً at a moment's notice .

مقطع توصيف البيانات The Metadata section

كل التفاصيل الإضافية التي يحتاجها الدوت نت للبحث ضمنه rummage through وذلك لمعرفة ما يخص مجمعك تظهر في هذا المقطع الأساسي.بعض من هذه البنود،عندما تأخذ مع بعضها ،تكّون كشف المجمع assembly's manifest(نوع مستند يشرح المجمع للعالم بالكامل).في القائمة التالية من عناصر توصيف البيانات metadata elements،أشرت إلى البنود التي تظهر في الكشف manifest:

إسم المجمع The name of the assembly

(جزء من الكشف)يتم تحديد هذا الجزء على تبويب التطبيق application لخاصيات المشروع project's properties.

رقم الإصدار(النسخة) للمجمع.The version number of the assembly

(جزء من الكشف)وهو عدد من أربع أجزاء للنسخة،كما في 1.2.3.4.،ربما تسآلت في يوم من الأيام عن كيفية وضع هذا العدد في مشروعك الخاص.سيتم الرد على صبرك في هذا الفصل وفي مقطع "المشروع"،عندما سوف أوضح ليس فقط طريقة ولكن اثنتين عن كيفية وضع رقم نسخة المجمع.

محتوى الاسم القوي Strong name content

(جزء من الكشفmanifest)يتضمن المفتاح العام للنشر publisher's public key.

إعدادات اللغة والثقافة Culture and language settings

(جزء من الكشف)وهو مفيد خاصة عندما تحتاج لإنشاء ملفات مصدرية خاصة باللغة .

جدولة ملف المجمع Assembly file listing

(جزء من الكشف).سترى مجمعات ملف مفرد فقط اسم ملف EXE أوDLL،ولكن بعض المجمعات يمكن أن تتضمن عدة ملفات في هذا المقطع. كل الملفات في مجمع يجب أن تظهر ضمن نفس الدليلdirectory ،أو في دليل تابع subordinateلملف المجمع الذي يحوي الكشف manifest.

معلومات الأنواع المصدرة Exported type information

(جزء من الكشف)بعض المجمعات "تصدر"بعض من أنواعها لأن تستخدم خارج التطبيق.فتفاصيل هذه الأنواع تظهر في هذا المقطع.

المراجع References

(جزء من الكشف،ولكن في المجمعات المتعددة الملفات،كل ملف سيحتوي قائمة خاصة به من المراجع)تتضمن وصف البيانات metadataجدولة لجميع المجمعات الخارجية external assemblies المرجعة(المشار إليها)بواسطة تطبيقك،فيما إذا كانت خاصة privateأو تظهر في GAC.تشير هذه القائمة إلى أي إصدار خاص specific version،ثقافةculture، والمنصة المستهدفة platformtarget للمجمع الخارجي الذي يتوقعه مجمعك .

معلومات الأنواع الداخلية Internal type information

(ليست جزء من المانفست manifest أو الكشف)كل الأنواع المعمولة(المبرمجة)في مجمعك يتم وصفها بالكامل ضمن وصف البيانات(الميتاداتاmetadata) أيضاً،يظهر هنا أي وصف بيانات إضافي additional metadata تم إضافته لأنواعك من خلال ميزة مواصفة الفيجوال بيسك Visual Basic's attribute feature.في المجمعات المتعددة الملفات،تظهر العناصر الخاصة بالمنانفست manifest-specific elements فقط في الملف الرئيسي"main" للمجمع .

المانفست (الكشف)هو مجموعة جزئية من الميتاداتا ضمن مجمعك.وإني أكره أن أقول أنها الجزء الأكثر أهمية للميتاداتا،ولكن هي كذلك.فالمنانفست هو تعبير عام لمجمعك،والطريقة الوحيدة التي يعلم فيها الدوت نت فيما إذا كان مجمعك شرعي(صحيح.legit).

حتى قبل ظهور الدوت نت،كانت الملفات القابلة للتنفيذ executablesوالمكتبية libraries تحتوي بعض "الميتاداتا،مثل رقم النسخة(الإصدار)للملف ،ولكن لم تكن هذه البيانات تستخدم لإدارة الوصول بين المكونات البرمجية،ولا حتى كانت منظمة في بطريقة عامة وقابلة للتوسيع(التمديدextensible).تشمل الميتاداتا في الدوت نت كل هذه المواصفات.إن وجود كل من لغة ميكروسوفت الوسيطة MSILوالميتاداتا metadataفي كل مجمع تجعل هذا الملفات قابلة للقراءة والفهم بشكل جيد.بالأدوات المناسبة With the right tools،حتى ولو كنت أبدو أني أفهمها ،وحتى لو استطعت،فأي شخص بإمكانه،وهذا ما يقود لمشكلة كبيرة.إن الشركات تستثمر الكثير من الوقت والنقود على الجهود التي تعمل على تطوير برمجياتها،ولا تريد من أي مبرمج متطفل أن يعكس هندسة كودها ويحصل على جميع أسراره وخوارزمياته السرية.ولمنع القراءة العرضية casual reading لأي من تطبيق الدوت نت.عملت ميكروسوفت وجهات أخرى third parties على تضمين المبهم(المشوش *obfuscators*)،وهي البرامج البرمجية التي تشفر(تدمج) محتوى أي مجمع بشكل كافي مما يجعل من الصعوبة بمكان لأي شخص من أن يفهمه،ولكن ليس بالنسبة لإطار عمل الدوت نت،سأتكلم أكثر حول التشويش obfuscation فيما بعد .

الانعكاسReflection

إنه يبدو شيء سيء أن يتمكن الناس من الوصول لمحتوى مجمع ما،وإنه لعظيم أن يتمكن الكود في مجمع من الوصول لنفسه.تتضمن الدوت نت مواصفة تدعى *reflection* تتيح لك تفحص محتويات مجمع ما.بشكل عام تستخدم هذه المواصفة للوصول إلى الميتاداتا metadataفي مجمعك الخاص،ولكنها تعمل أيضا مع أي مجمع متاح. تظهر معظم الميزات المتعلقة بالانعكاس في فضاء الأسماء System.Reflection namespace.

من خلال الانعكاس بإمكانك استخراج تقريباً كل شيء مخزن في الميتاداتا metadataلمجمع ما،ومن ضمنها تفاصيل على كل الأنواع،أعضاءها،وحتى الوسيطات المضمنة مع أعضاء الدوال .وهذا ما يجعل التشويش obfuscationهاماً جداً بالنسبة للبائعينvendors ،بين لغة ميكروسوفت الوسيطة المترجمة compiled MSIL والميتاداتاmetadata ،بإمكانك افتراضياً إعادة توليد الكود المصدري بالكامل لتطبيق ما فقط من ملفه القابل للتنفيذ.ولن يكون بتلك الصعوبة بالنسبة لشخص ما متمرس في استعادة الكثير منه ضمن الفيجوال بيسك أوC# .

المجمعات والتطبيقات Assemblies and Applications

تطبيقات الدوت نت(ملفات تنفيذية EXE)هي حالة(نسخةinstance)من مجمع،ولكن يمكن لتطبيق مفرد أن يحوي على عدة مجمعات،في الحقيقة،إنه غالباً ما يكون كذلك. قمت بكاتبة برنامج صغير يستخدم الانعكاس لجدولة جميع المجمعات list all assemblies actively التي تم استخدامها فعلياً actively بواسطة البرنامج نفسه،وأعطيته الاسم الافتراضي *WindowsApplication1*.وعندما شغلت البرنامج على نفسه،ولد القائمة التالية:

mscorlib
Microsoft.VisualStudio.HostingProcess.Utilities
System.Windows.Forms

System
System.Drawing
Microsoft.VisualStudio.HostingProcess.Utilities.Sync
Microsoft.VisualStudio.Debugger.Runtime
vshost
System.Data
System.Deployment
System.Xml
System.Core
System.Xml.Linq
System.Data.DataSetExtensions
Microsoft.VisualBasic
WindowsApplication1
System.Runtime.Remoting

كما ترى 17مجمع من ضمنها *WindowsApplication1* البرنامج الرئيسي main program. معظم المجمعات DLLs مزودة من قبل إطار العمل-framework supplied.من أجل ميكروسوفت.فيجوال بيسك Microsoft.VisualBasic المجمع هو *Microsoft.VisualBasic.dll*،ومن أجل النظام System،المجمع هو *System.dll*.

كل المجمعات (ما عدا مجمع البرنامج الرئيسي)هي متشاركة المكتبات libraries من GAC.يمكن للتطبيق أيضاً من دعم مجمعات خاصة private assemblies محملة من ملفات DLL محلية .

يحمِّل إطار عمل الدوت نت NET Framework. هذه المجمعات لي بشكل آلي عندما يشتغل *WindowsApplication1*،إنه يستكشف أي منها فيما إذا كان هناك حاجة إليه لكي يتم تحميله وذلك بالنظر في المانفست لـ *WindowsApplication1*.عندما حمل إطار العمل كل مجمع،فإنه يتفحص ليرى فيما إذا كانت هذه المجمعات بدورها تحتاج لمجمعات إضافية ليتم تحميلها،وهكذا.تطبيق بسيط وحيد أصبح أرض تفريغ للمجمعات من كافة أنحاء GAC.ولكن هذا جيد،بما أن هدف الدوت نت هو إدارة هذه المجمعات جميعها.

المجمعات وفضاء الأسماء الخاص My The My Namespace and Assemblies

إن إطار العمل مع الآلاف من فئاته،يحتوي الكثير من المنطق المشحون(المحزم) packaged logicوالذي بإمكاني استخدامه في برنامجي الخاص.ولكنني لا أملك تذكر المجمعات العديدة وفئائهم الكثيرة.وقد يأخذ الكثير من الوقت لتطوف ضمن توثيق مكتبة فئات إطار العمل FCL documentation. مع الكثير من الفئات المتاحة،فإني في بعض الأحيان أرتعد shudderعندما أفكر بالجهد الذي سأبذله لإيجاد الفئة الصحيحة فقط أو الميزة التي أحتاجها لإتمام بعض مهام التطوير(البرمجة) .

لحسن الحظ،لست الوحيد الذي يفكر بهذه الطريقة،فميكروسوفت تتفق معي.تاريخياً ،كان مبرمجي فيجوال بيسك محميين sheltered من التعقيدات في تطوير تطبيق ويندوز.وليس هذا ما يحتاجون أن يكونوا عليه،فنحن جميعاً نعرف أن مطوري الفيجوال بيسك بشكل عام مفضلين على الباقي.ولكن "كان شعار الفيجوال البيسك"التحاور على:جعل تطوير ويندوز أسرع وأسهل *Make Windows Development Fast and Easy*. واستدعاء بعض الطرق المقتصرة esoteric والراسخة ضمن أجزاء bowelsنظام فضاء الأسماء System namespace لتحصل على القسم الرئيسي من البيانات وهو ليس بالسهل ولا حتى بالسريع.

ولاستعادة بعض البساطة والسرور التي كانت متاحة سابقاً في الفيجوال بيسك،عملت ميكروسوفت على تقديم استدعاء فضاء الأسماء My في إصدار 2005من اللغة.يضم استدعاء فضاء الأسماء My الكثير من الميزات المفيدة من كامل FCL،ويعمل على تنظيمها في طبقات هرمية hierarchyأكثر صغراً من أجل البساطة والسهولة في الوصول.ولقد قمت بذكر باختصار Myفي الفصل الأول،ولكن حالياً من الجيد أخذ نظرة أقرب لما يفعله.

يبدو pretend namespace مستدعي فضاء الأسماء Myمشابه كثيراً لأي فضاء أسماء أخر،مثل System، System.Reflection،وSystem.Windows.Forms. ولكنه في الحقيقة ليس فضاء أسماء-فهو مستدعي pretend.

لسبب ما،لاتستطيع استخدام الكلمة المحجوزة Imports لإنشاء اختصار للتفرعات(التشعبات)ضمن هيكله الهرمي.على الرغم من أن بعض المقاطع من هرمية التسلسل هي ديناميكية ،فهي تتغير كما يتغير مشروعك،يجدول الجدول التالي العقد nodesالرئيسية للتسلسل الهرمي .

التفرع Branch	الميزات المتاحة Available features
My.Application	يوفر معلومات حول التطبيق الحالي،ومن ضمنها إعدادات الخيارات الإقليمية وطرق النشر .
My.Application.Info	يعطي معلومات أكثر حول التطبيق ومجمعاته،ومن ضمنها الاسم nameوالنسخة version.
My.Application.Log	يسمح لك توليد متتبع ومخرجات تسجيل دخول لأغراض تسجيل الدخول المسجلة،ويستخدم فقط مع تطبيقات العميل
My.Computer	يوفر إمكانية الوصول إلى المصادر العامة المتواجدة على الكمبيوتر المحلي.
My.Computer.Audio	يشغل أصوات النظام وأصوات أخرى محددة من خلال سماعات الكمبيوتر .
My.Computer.Clipboard	يستخلص البيانات من حافظة النظام،ويتيح لك إضافة بيانات خاصة بك للحافظة في تنوع محدد مسبقاً وتنسيقات خاصة.
My.Computer.Clock	يحصل على تاريخ النظام الحالي والوقت المعروض بطرق متنوعة.
My.Computer.FileSystem	يوفر أدوات لتفحص ومعالجة الملفات والأقراص على أنظمة الملفات filesystems المحلية أو الشبكية .
My.Computer.FileSystem. SpecialDirectories	يشير إلى مجلدات ويندوز الخاصة مثل المستندات*Documents*، سطح المكتب*Desktop*، والمجلدات المؤقتة *Temp*.
My.Computer.Info	يوفر معلومات حول نظام التشغيل المثبت ومصادر النظام المحلية الأخرى .
My.Computer.Keyboard	يعرض الحالة الحالية من لوحة المفاتيح ومفاتيحها .
My.Computer.Mouse	يجعل العديد من الخصائص لفارة الكمبيوتر المحلي متاحة(ممكنة).
My.Computer.Network	يبلغ عن الشبكة المتاحة،ويوفر ميزات للتفاعل مع تلك الشبكة.
My.Computer.Ports	يتيح لك التفاعل مع مداخل(منافذ أو شقوق) النظام التسلسلية system's serial ports.
My.Computer.Registry	يقرأ ويكتب المفاتيح keysوالقيم valuesفي المسجل registry.
My.Forms	يجلب تجمع ديناميكي لكل النماذج المعرفة في التطبيق.وهذه العقدة متاحة فقط لتطبيقات نماذج ويندوز.
My.Log	يسمح لك من توليد متتبع ومخرجات تسجيل دخول للأغراض التسجيلية المسجلة.ويستخدم مع تطبيقات ASP.NET فقط .
My.Request	هذا الكائن مشابه لمدخل الصفحات الفعال Active Server Pagesالقديم الكائن Request.وهو متاح فقط من أجل تطبيقات ASP.NET.
My.Resources	يوفر إمكانية وصول ديناميكي لتطبيق مخصص أو مصادر محلية مخصصة ومضمنة في التطبيق .

My.Response	هذا الكائن مشابه لمخدم الصفحات الفعال Active Server Pages القديم الكائن Response.وهو متاح فقط من أجل تطبيقات ASP.NET
My.Settings	يوفر إمكانية وصول ديناميكي لنظام إعدادات التطبيق application settings system.
My.User	يحدد مستخدم ويندوز Windows user الحالي،ويتضمن معلومات توثيق authentication information.
My.WebServices	يجلب تجمع لخدمات الويب المتاحة available web services لكي تستخدم في التطبيق.وهذه العقدة غير متاحة في تطبيقات ASP.NET.

يتضمن فضاء الأسماء الكثير من الميزات التي ستستخدمها بانتظام،من ضمنها الوصول لرقم نسخة التطبيق،بدلاً من كتابة *System.Reflection. whatever*
للحصول على رقم النسخة التابعة للمكون الرئيسي"major"،بإمكانك الآن كتابة فقط:

```
My.Application.Info.Version.Major
```

هل تحتاج لقائمة من المجمعات،ولكنك كسول جداً لأن تكتب الكلمة *Reflection* ؟طيب جرب التالي:

```
My.Application.Info.LoadedAssemblies
```

هل تحتاج لمعرفة الوقت الحالي بالنسبة للتوقيت العالمي أو في بريطانيا؟ جرب التالي:

```
My.Computer.Clock.GmtTime
```

بإمكانك الاتصال بالشبكة المحلية local area network:كما يلي:

```
My.Computer.Network.IsAvailable
```

من يشغل هذا الكمبيوتر على أية حال؟:

```
My.User.Name
```

تتضمن أيضاً الفيجوال بيسك الأمر "قتل Kill" الذي يسمح لك من حذف الملفات.تزيل الطريقة My.Computer.FileSystem.Delete أيضاً الملفات،ولكنها توفر خيارات إضافية من ضمنها أنها تسمح لك من إرسال ملف إلى سلة المحذوفات Recycle Bin بدلاً من فقدانها بالكامل(للأبد).

التوجيهات(التعليمات) والمجمعات Directives and Assemblies

التوجيهات هي عبارات فيجوال بيسك،ولكنها فيما بعد،تصبح ليست كذلك.التوجيهان المفتاحيان #Const و#If يوفران تعليمات للمترجم عن كيفية معالجة مقطع من الكود المصدري للفيجوال بيسك.(التوجيه الثالث#Region، يساعد على جلب الكود المصدري ضمن الفيجوال أستوديو بشكل مرئي،ولكن ليس له تأثير على المترجم أو على التطبيق المترجم النهائي) باستخدام التوجيهات،تستطيع إخبار المترجم على تضمين أو إخراج قطعة خاصة من الكود المصدري من المشروع النهائي.لذلك هي ليست عبارات كود مصدري للفيجوال بيسك حقيقية،ولكنها متاحة فقط في الفيجوال بيسك.ولكن لماذا تريد تضمين أو إخراج كود في تطبيق ما؟حسناً،لعدة أسباب،على سبيل المثال،النسخ المتعددة لتطبيقك،بالاعتماد على بعض الشروط،فيمكن أن تشتري نسخة"سريعة express " ونسخة "محترفة professional " لمنتج ما.معظم الكود يكون متشابه لكلا النسختين.وأيضاً،يمكن للنسخة الاحترافية أن تتضمن ميزات غير متاحة في النسخة السريعة،أيضاً،النسخة السريعة يمكن أن تتضمن تقديم بسيط لميزة لديها استخدام أكثر تعقيد في الطبعة الاحترافية .
بعض المنتجات البرمجية تحقق هذه الحاجة باستخدام شروط فيجوال بيسك القياسية.

```
If (professionalVersion = True) Then
        ShowWhizBangFeatures()
    Else
        ShowLaughableFeatures()
    End If
```

إن هذا يعمل بشكل جيد،ولكن مايزال التطبيق السريع يحتوي على الميزات المحسنة جميعها.حيث أنه فقط لايمكن الوصول إليها من أي كود،ولما يتم تضمينه على سيدي التنصيب؟إذا استخدمت التوجيهات،تستطيع تخفيض mark down تلك المشكلة كأنها حلت.تستخدم التوجيهات التعابير الشرطية، تشبه كثيراً الشرط professionalVersion = True في مقطع الكود السابق.ولكن يتم تعريفها بواسطة العبارة #Const،ويتم استدعاء ثوابت المترجم *compiler constants*.

```
#Const fullVersion = True
```

تعرِّف هذه العبارة ثابت مترجم منطقيBoolean compiler constant. ويمكن للثابت أن يستخدم فقط مع التوجيهات،إذا حاولت استخدام fullVersionفي عبارة فيجوال بيسك قياسية،فإن المترجم سيتذمرcomplain. ولكن سيعمل بشكل جيد في توجيه #If .

```
#If (fullVersion = True) Then
ShowWhizBangFeatures( )
#Else
      ShowLaughableFeatures()
#End If
```

إن هذا الكود يبدو مشابه كثيراً لمقطع الكود السابق،ولكن مع الإشارات # المضافة.فهما يبدوان نفس الشيء ولكنهما ليسا كذلك.مع سلاسة عبارة If،فإن الكود التالي سيتم ترجمته في التطبيق النهائي:

```
If (professionalVersion = True) Then
        ShowWhizBangFeatures()
    Else
        ShowLaughableFeatures()
    End If
```

نعم،كامل مقطع الكود.ولكن مع الموجهات،ما يتم تضمينه في التطبيق المترجم يعتمد على قيمة fullVersion.فإذا كانت fullVersion صوابTrue فإن التالي سيحصل على الترجمة في التطبيق المترجم:

```
ShowWhizBangFeatures()
```

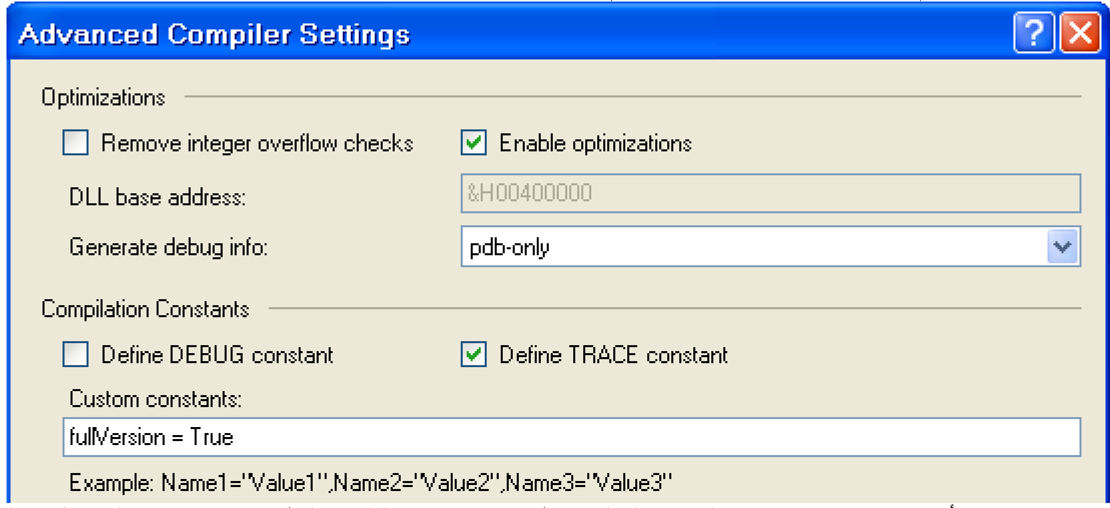
أما العبارات الأربع الباقية ستذهب،أي ستذهب أدراج الرياح،وكأنها لا توجد أبداً.ولكن في هذه الحالة،هذا شيء جيد.فالهدف كان الحصول على نسخة من المجمع المترجم خالٍ(مجرد)من الكود الغير مطلوب(الغير مرغوب)،وهذا ما حصل.
لوضع ثابت المترجم fullVersionلتوليد نسخة كاملةfull version، فإنك تضمن هذا السطر عند أعلى ملف الكود المصدري والذي يتضمن مقاطع الكود#Ifالشرطية:

```
#Const fullVersion = True
```

عندما تكون جاهز لتوليد نسخة "سريعة"،فقط غير كل هذه الأسطر إلى نظيرها "خطأ False ":

```
#Const fullVersion = False
```

شيء ما ،يبدو أن تغيير هذا السطر في كل ملف كود مصدري عمل كثير،وهو كذلك.وماذا سيحدث إذا ما نسيت وضع واحد منها إلى النسخة المناسبة؟ بإمكاني إخبارك ،أن هذا ليس بالجيد.لذلك توفر الفيجوال أستوديو العديد من الطرق لوضع ثوابت المترجم فوراً،وتطبيقهم لك كجزء من التطبيق. الطريقة الأكثر شيوعاً لعمل مثل هذا من خلال لوحة المترجم Compileالتابعة لصفحات خاصيات المشروع project properties (شاهد الشكل التالي).أنقر على زر خيارات المترجم المتقدمة Advanced Compile Options،ومن ثم اعمل على إضافة ثوابت المترجم الشاملة الخاصة بك لحقل "ثوابت مخصصة Custom constants "



والآن بإضافة إما fullVersion = True أو fullVersion = False إلى هذا الحقل،تستطيع بناء نسخ مختلفة من التطبيق.يوفر مترجم الفيجوال بيسك Visual Basic compiler أيضاً ميزات تسمح لك من تثبيت صيغ ترجمة مختلفة different compile scripts لمشروعك.ولن أدخل ضمنها في هذا الكتاب،ولكن بإمكانك قراءة معلومات عنها من مستندات فيجوال أستوديو ضمن أدوات MSBuildإذا كنت تريد هذا المستوى من الأدوات.

يمكن لثوابت المترجم compiler constants أن تكون أعداد numbersأو سلاسل نصية strings بالإضافة إلى الثوابت المنطقيةBooleans. بيئة تطوير الفيجوال أستوديو تعين أيضاً بعض ثوابت المترجم من أجلك. فالثوابت DEBUG وTRACE هي إما صح أو خطأ بالاعتماد على صندوق اختيار " Define DEBUG constant " و" Define TRACE constant " الذي يظهر في الشكل السابق.

مشروع Project

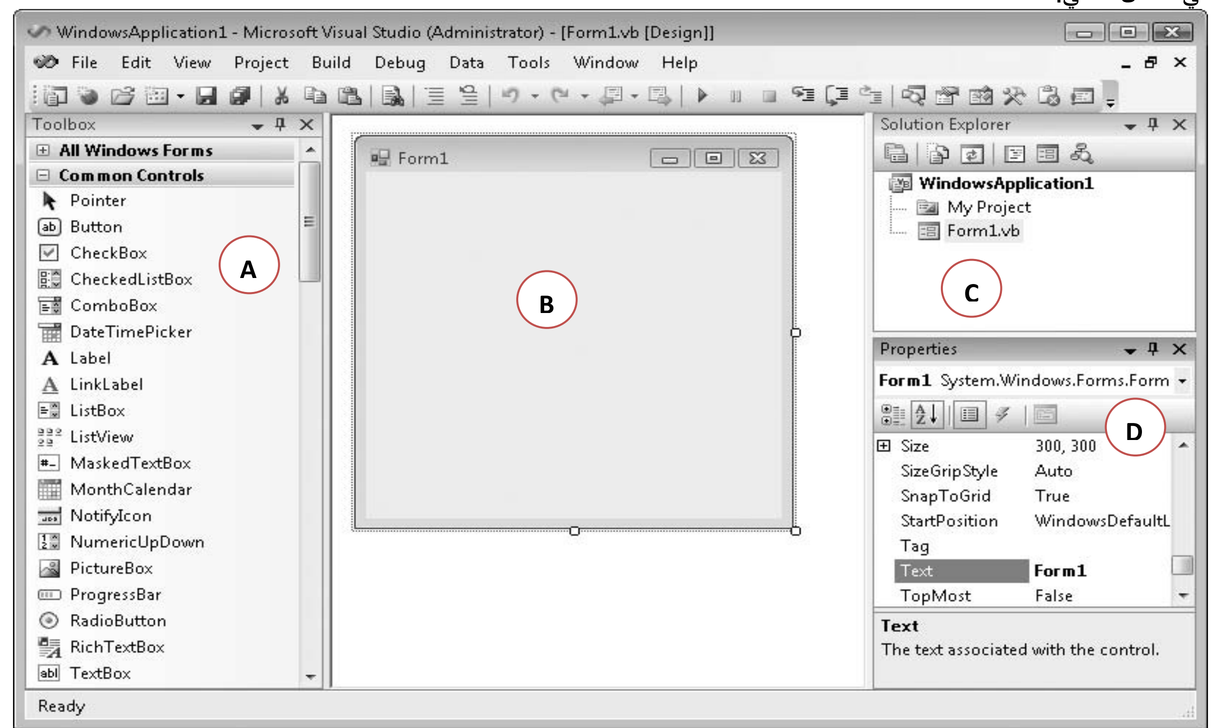
في مشروع هذا الفصل رسمياً سنتابع كتابة كود مشروع المكتبة،سنبدأ بشيء ما بسيط:بناء المعلومات الأساسية حول التطبيق والفورم التابع له،بما في ذلك رقم النسخة version number.

هدفنا هو الوصول لفورم ساحرة تنقل conveysالمعلومات الأساسية حول البرنامج.

مثل أي تطبيق فيجوال بيسك لويندوز،إنشاء هذه الفورم يتضمن خطوتين☹1)إضافة الأدوات للفورم،و(2)كتابة الكود الخاص بها.

إضافة الأدوات Adding Controls

إذا كان هناك مساحة يتفوق excels فيها الفيجوال بيسك،فهي إنشاء الفورم.يمكن أن يتم إنشاء البرامج بواسطة السحب والإسقاط للأدوات الجاهزة على سطح الفورم الجاهزة.وهذا كله يتم عمله من ضمن بيئة تطوير الفيجوال أستوديو المتكاملة(Integrated Development Environment (IDE المريحة والمناسبة.كما هو مبين في الشكل التالي.



تتضمن بيئة التطوير المعروضة أربع مساحات رئيسية،والتي وسمتها بالأحرف المبينة في الشكل في الأعلى:

A. صندوق الأدوات:*toolbox*

وهو قائمة بالأدوات المضمنة وليس عرض للأدوات فقط.

B. سطح الفورم The form surface

ضع أي أداة هنا ليتم عرضها على واجهة المستخدم.والفورم " ما تراه هو ما تحصل عليه في النهاية WYSIWYG "لذلك فإنك تستطيع رؤية التصميم النهائي بالشكل الذي تصمم الفورم به.

C. مستكشف الحلول The Solution Explorer

كل الملفات المتعلقة بمشروعك تظهر هنا.من أجل المشروع الحالي سترى فقط المدخلة My Project و المدخلة form،*Form1.vb*. يوجد عملياً ملفات أكثر.إذا نقرت على الزر الثاني من اليسار عند أعلى مستكشف الحلول،فإنه سيريك ملفات إضافية،ومعظم هذه الملفات يتم إدارتها بواسطة الفيجوال أستوديو من أجلك.

D. نافذة الخصائص The Properties panel

عندما تختار أداة على سطح الفورم،أو سطح الفورم نفسها،أو أي بند في مستكشف الحلول،ستظهر خصائص البند المختار في هذه المنطقة.فتستطيع تغيير إعدادات العديد من الخصائص بكتابة إعدادات جديدة.بعض الأدوات تتضمن أدوات خاصة لمساعدتك في وضع قيمة الخاصية.

سنعمل على إضافة ثمانية أدوات "عنوان labels "،وثلاث عناصر "شكل shape "و"سطرline "،ارتباطي انترنت تشعبيينweb-style hyperlinks،زر button،وصورة pictureلسطح الفورم.

اعمل على إعداد الفورم وذلك بضبط الخصائص التالية ،انقر على سطح الفورم،ومن ثم عدل قيم هذه الخصائص باستخدام نافذة الخصائص:

الخاصيةProperty	الإعداد Setting
(Name)	AboutProgram
ControlBox	False
FormBorderStyle	FixedDialog
Size	440, 311
StartPosition	CenterScreen
Text	حول مشروع المكتبة

بعدها،اعمل على إضافة ثمانية أدوات "عنوان Label " لسطح الفور من صندوق الأدوات باستخدام الأداة " Label".واستخدم القائمة التالية لوضع الخصائص بالنسبة لكل أداة عنوان.

اسم أداة العنوان الافتراضي	الخاصيات التي سنعمل على تعديلها
Label1	(Name): ProgramName AutoSize: True Font/Bold: True Location: 136, 16 Text:مشروع المكتبة
Label2	(Name): ProgramVersion AutoSize: True Location: 136, 32 Text:إصدار س.ع التنقيح ص
Label3	(Name): LicenseInfo AutoSize: False Location: 136, 48 Size: 280, 32 Text:غير مرخص
Label4	(Name): DevelopedBy AutoSize: True Location: 136, 88 Text:تمت البرمجة بواسطة
Label5	(Name): DeveloperName AutoSize: True Location: 160, 112 Text: MHM
Label6	(Name): DeveloperBook AutoSize: True Location: 160, 128 Text: Programming Visual Basic 2008
Label7	(Name): DeveloperProject AutoSize: True Location: 160, 144 Text: In-book Project
Label8	(Name): CompanyCopyright AutoSize: True Location: 136, 208 Text: Copyright (c) 2009 by MHM.

لنضيف بعض الأسطر والمقطع الملون للفورم.تتضمن الفيجوال بيسك6 أدوات أشكال مميزة للأسطرlines،المستطيلاتrectangles،القطع الناقص ellipses والتي تستطيع تطبيقها مباشرة لسطح الفورم.أما الدوت نت فإنها لم تعد تتضمن هذه البنود،عليك إضافتها بمساعدة الكود المصدري الذي يستخدم أوامر الرسم المخصصة.ولكن نستطيع محاكاة الأسطر والمستطيلات باستخدام أداة "عنوان Label "قياسية، خالية sansمن النص .

اسم أداة العنوان الافتراضي	الخاصيات التي سنعمل على تعديلها
Label9	(Name): VersionDivider AutoSize: False BackColor: Black Location: 136, 80 Size: 280,1 Text:(لا تضيف أي نص)
Label10	(Name): BackgroundSide

اسم الأداة	الخاصيات المعدة
	AutoSize: False BackColor: White Location: 0, 0 Size: 120, 296 Text:(لا تضيف أي نص)
Label11	(Name): BackgroundDivider AutoSize: False BackColor: Black Location: 120, 0 Size: 1, 296 Text:(لا تضيف أي نص)

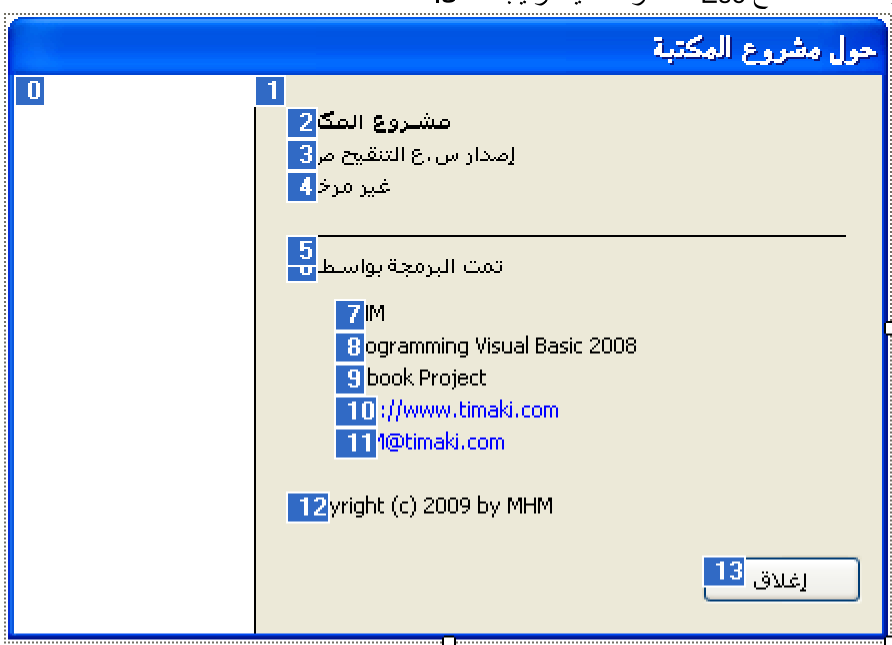
إذا كانت أداة الأداة Label11تحجب الصورة،انقر يمين عليها ومن القائمة المنسدلة اختر "إرسال إلى الخلفية Send To Back ".
أداة"عنوان الوصلة LinkLabel"مشابهة لأكثر أدوات العنوان الأساسيةLabel، ولكن بإمكانك تضمين "وصلات links " في النص،مقطع قابل للنقر clickable sections والذي يكون مشابه للوصلات على صفحات الانترنت.وسنستخدمها لعرض موقع ويب وعنوان أيميل.أضف أداتي " LinkLabel" للفورم واستخدم الإعدادات التالية لتركيب خاصيات الأدوات.

اسم أداة عنوان الوصلة الافتراضي	الخاصيات المعدة
LinkLabel1	(Name): CompanyWeb AutoSize: True LinkBehavior: HoverUnderline Location: 160, 160 Text: http://www.mhm.com
LinkLabel2	(Name): CompanyEmail AutoSize: True LinkBehavior: HoverUnderline Location: 160, 176 Text: MHM@yahoo.com

الأداة الأخيرة التي سنضيفها هي زر والذي يسمح للمستخدم من إغلاق الفورم.أضف أداة زر Button للفورم مع الخاصيات التالية:

اسم الزر الافتراضي	الخاصيات التي سيتم إعدادها
Button1	(Name): ActClose DialogResult: Cancel Location: 344, 240 Size: 80, 24 Text:إغلاق

يمكن أن يتم تركيب الفورم بحيث أن ضغط المفتاح Escيطلق أداة الزر على الفورم،وكأن المستخدم ينقر على الزر نفسه عوضاً عن ضغط المفتاح Esc.لعمل هذا ،انقر على سطح الفورم،وضع خاصيتها CancelButtonإلى ActClose.لقد أجلنا هذه الخطوة حتى يتم إضافة الزر بشكل فعلي للفورم،إن الخاصية CancelButton لن تسمح أن يتم إعدادها إذا لم يكن هناك زر موجود.حسناً،والآن ستبدو الفورم مناسبة،الشيء الأخير الذي أحب أن أعمله هو وضع ترتيب التنقل،الترتيب الذي يمكن للمستخدم أن يصل به لكل حقل على الفورم عند الضغط على مفتاح "التنقل Tab " من لوحة المفاتيح.ولتحرير ترتيب التنقل،اختر سطح الفورم و اختر عرض View >> الأمر "ترتيب التنقل Tab Order" من القائمة.كل أداة على الفورم التي يمكن أن تأخذ قيمة ترتيب تنقل سيكون لديها ترتيب تنقل بجانب كل منها.
انقر على كل عدد أو أداة بالترتيب حتى تحصل على الترتيب الذي تريده.(شاهد الشكل التالي لترى كيف رتبت الأدوات).أخيراً،اختر عرض View>> أمر ترتيب التنقل Tab Order مرة أخرى،أو اضغط المفتاح Esc،لمغادرة عملية ترتيب التنقل.



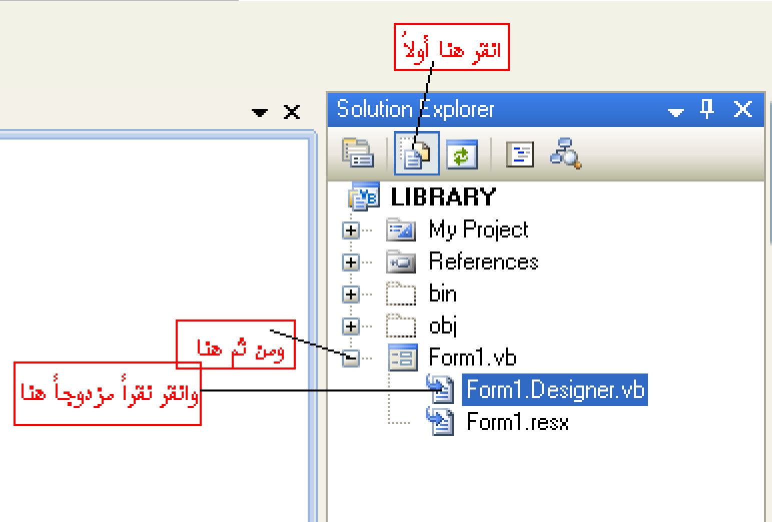
تستطيع أيضاً وضع ترتيب التنقل لكل أداة بتعديل خاصيتها TabIndex،ولكن الطريقة الأسرع التي هي التي قمنا بها للتو.

إضافة كود إلى الفورم Adding the Code to the Form

والآن،حان الوقت المناسب لإضافة بعض كود فيجوال بيسك الحقيقي.لنأخذ نظرة سريعة على مستكشف الحلول،انقر زر "أظهر جميع الملفاتShow All Files " عندما تظهر جميع الملفات انقر على إشارة الزائد بجانب Form1.vb،وأخيراً،انقر مزدوج على Form1.Designer.vb (شاهد الشكل التالي).

نظراً لكونها أكثر من 200سطر من الكود المصدري،فلن أطبعها هنا،ولكن انظر إليها بشكل عام،فعندما تعمل على سحب وإسقاط أدوات على الفورم وتعدل خصائصها،تعمل الفيجوال أستوديو على تعديل هذا الملف من أجلك.وهو جزء من فئتك للنموذجform's class (جميع النماذج هي فئات يتم اشتقاقها من System.Windows.Forms.Form).بإمكانك أن تعلم ذلك بواسطة الكلمة المحجوزة Partialعند الأعلى .

```
Partial Class AboutProgram
    Inherits System.Windows.Forms.Form
```



يحدث معظم الفعل في الإجراء InitializeComponent.عندما تنتهي من إلقاء نظرة شاملة عليه ككل أغلق مصمم الكود وارجع إلى سطح الفورم.
لجعل نموذجنا نموذج حقيقي وممتع،نحتاج لعمل ثلاث أشياء:

- إظهار رقم النسخة الحقيقي للتطبيق.وسوف يتم تحديده وعرضه عندما تظهر الفورم للمرة الأولى.
- الذهاب إلى موقع الانترنت المناسب أو متلقي الايميل المناسب عند النقر على لوحة الوصلة.وهذه الأحداث يتم معاملتها في حالة الاستجابة لفعل المستخدم.
- إغلاق الفورم عندما ينقر المستخدم على زر الإغلاق. وهو أيضاً حدث مقاد من قبل المستخدم.

لنبدأ بالأسهل ،إغلاق الفورم،فالأحداث هي مقاطع من الكود يتم معالجتها في حالة الاستجابة لشيء ما يحدث،وعلى الأغلب إجراء مستخدم مثل النقر بالفارة.كل الإجراءات التي نريد أن ننجزها على هذه الفورم ستكون رد فعل (استجابة) على الحدث المطلق(المحرر). الطريقة الأسهل للوصول إلى الحدث الافتراضي للأداة هو بالنقر المزدوج عليها.جرب هذا الآن على الزر.عندما تنقر مزدوج على الزر فإن بيئة التطوير ستفتح محرر الكود(الكود المصدري) ،ستضيف معالج حدث فارغ(الإجراء الفرعي ActClose_Click)

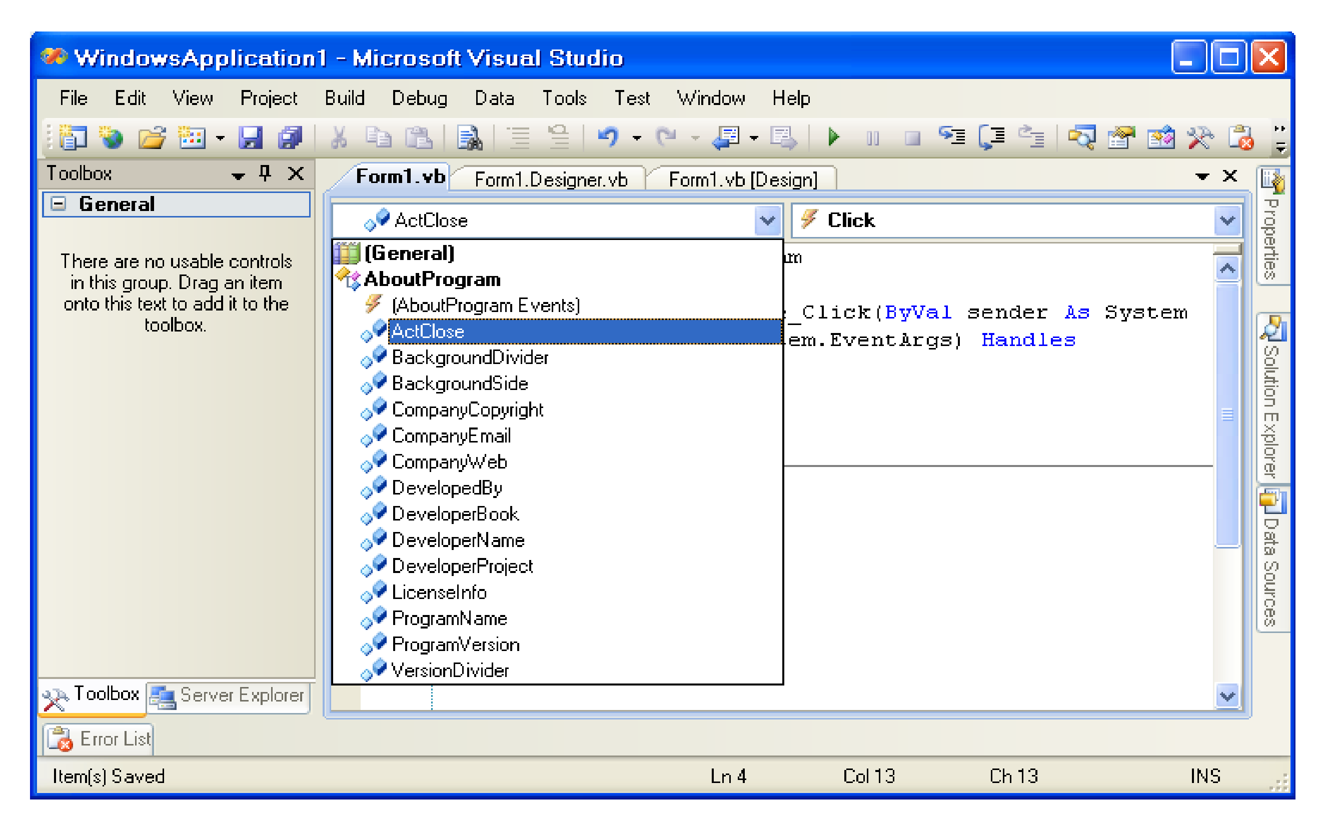
```
Public Class AboutProgram
    Private Sub ActClose_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ActClose.Click

    End Sub
End Class
```

كل حدث معتمد على النماذج(وفي الحقيقة ،معظم أنواع الأحداث الأخرى)في الدوت نت لديها نسبياً نفس ا لمعاملات النسبية☹1)المعامل النسبي senderوالذي يشير إلى الكائن الذي أطلق هذا الحدث،و(2) المعامل النسبي e،والذي يسمح لـ senderمن التزويد بأي معلومات إضافية يمكن أن تكون مفيدة في الحدث.في هذه الحالة، المعامل النسبي senderسيشير إلى الزر ActClose،بما أنه الكائن الذي سيولد حدث النقرClick. وحدث نقر الزر ليس لديه أي معلومات إضافية متاحة،لذلك e هو نوع الكائن الافتراضيSystem.EventArgs ، والذي يكون نسبياً حاجز مكان placeholder،وهو الكائن الذي منه كل أنواع المعاملات النسبية eالأكثر أهمية تستمد (تُشتق منه).
اسم معالج الحدث هذا هو ActClose_Click،ولكن إذا كنت تريد تغيره إلى أي شيء تريد ،فإنه لن يفقد أي شيء.ولكن عليك أن تحفظ التعبير(عالج نقر الزر.Handles ActClose.Click)سليمةintact . وهذا هو الجزء الذي يربط معالج الحدث event handler بالحدث الفعلي.
والكود اللازم لإغلاق الفورم بسيط جداً.أدخله الآن،بكتابته مباشرة:

```
Private Sub ActClose_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ActClose.Click
        Me.Close()
    End Sub
```

تقول هذه العبارة Me.Close()،(إني نموذج /كائن المشروع حول البرنامج،وآمر نفسي بالإغلاق.)إذ شغلت البرنامج الآن(اضغط F5)،إنك تغلق الفورم بالنقر على الزر "إغلاق".بما أن نموذج المشروع هو الفورم الوحيد في التطبيق،فإن إغلاقه يؤدي بشكل آلي إلى إنهاء كامل التطبيق.
ارجع إلى البند الثاني ،وصلات الانترنت.بإمكانك العودة إلى سطح الفورم والنقر المزدوج على كل لوحة وصلة لإنشاء معالج حدث لكل أداة لوحة وصلة افتراضي(في هذه الحالة،حدث LinkClicked).ولكن بإمكانك أيضاً إضافة الإجراء الفرعي لمعالج الحدث من ضمن المحرر،إما بكتابة الكود نفسه(وهذا غير مناسب) أو باستخدام قائمتي الانسدال فوق نافذة المحرر(شاهد الشكل التالي)



تظهر قائمة اسم الفئة(الكائن) في الجهة اليسارية.اختر أي مدخلة من هذه القائمة،واختر الحدث المناسب من القائمة اليمينية(قائمة اسم الطريقة Method Name).ولإضافة قالب معالج حدث لحدث CompanyWeb's LinkClicked،اختر أولاً CompanyWebمن قائمة اسم الفئة اليسارية،ومن ثم اختر من القائمة اليمينية (قائمة اسم الطريقة).وبالتالي سيظهر مقطع الكود التالي:

```
Private Sub CompanyWeb_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles CompanyWeb.LinkClicked

    End Sub
```

قائمة المعاملات النسبية لهذا القالب نوعاً ما أكثر إثارة،حيث أن المعامل النسبي eهو كائن من نوع: System.Windows.Forms.LinkLabelLinkClickedEventArgs.تسمح لك الأداة LinkLabel من أن يكون لديك عدة وصلات انترنت في أداة وحيدة، موزعة interspersed بين نصوص نظامية.للمعامل النسبي خاصية "الوصلة Link " والتي تخبرك أي من الوصلات في الأداة تم النقر عليها من قبل المستخدم.بما أن لوحاتنا labelsلها فقط وصلة وحيدة،فلن نعاني في اختبارها.سنظهر فقط صفحة الويب مباشرةً في أي وقت يتم النقر على الوصلة.

```
Private Sub CompanyWeb_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles CompanyWeb.LinkClicked
        ' ----- Show the company web page.
        Process.Start("http://www.mhm.com")
    End Sub
```

الكائن Processهو جزء من فضاء الأسماء System.Diagnostics،و Startهي واحدة من أعضاءه المتشاركة والتي تتيح لك من تشغيل تطبيقات ومصادر خارجية.إنك تمرر له أي عنوان انترنت URLمناسب وهو سيشغله باستخدام مستعرض المستخدم الافتراضي أو التطبيق الخاص بعناوين الانترنتURL .لنجربه مرةً أخرى مع حدث CompanyEmail's LinkClicked.أضف قالبه بأي طريقة تختارها ومن ثم اكتب أو أدخل الكود الذي يشغل رسالة جديدة لعنوان الايميلemail .

```
Private Sub CompanyEmail_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles CompanyEmail.LinkClicked
        ' ----- Send email to the company.
Process.Start("mailto:MHM@yahoo.com")
    End Sub
```

الحدث الأخير الذي سنصممه هو واحد من الأحداث الأولى والتي يتم استدعاءها في دورة حياة الفورمform: الحدث تحميل Load.ويتم استدعاءه تماماً قبل أن تظهر الفورم على الشاشة.انقر مزدوج في أي مكان على سطح الفورم وهذا سينشئ قالب معالج الحدث الخاص بحدث التحميل.(بإمكانك استخدام نافذة محرر الكود ومن قائمة الفئة في الجهة اليسارية أعلى نافذة المحرر اختر الفورم ومن قائمة الطرق في أعلى اليمين اختر الطريقة تحميلLoad).

لنضيف لهذا الحدث الكود الذي سيعرض رقم النسخة الصحيح،باستخدام My.Application.Info.Version،وهي نسخة (حالة) من الفئة System.Version.

```
Private Sub AboutProgram_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
        ' ----- Update the version number.
        With My.Application.Info.Version
            ProgramVersion.Text = "Version " & .Major & "." &.Minor & " Revision " & .Revision
        End With
    End Sub
```

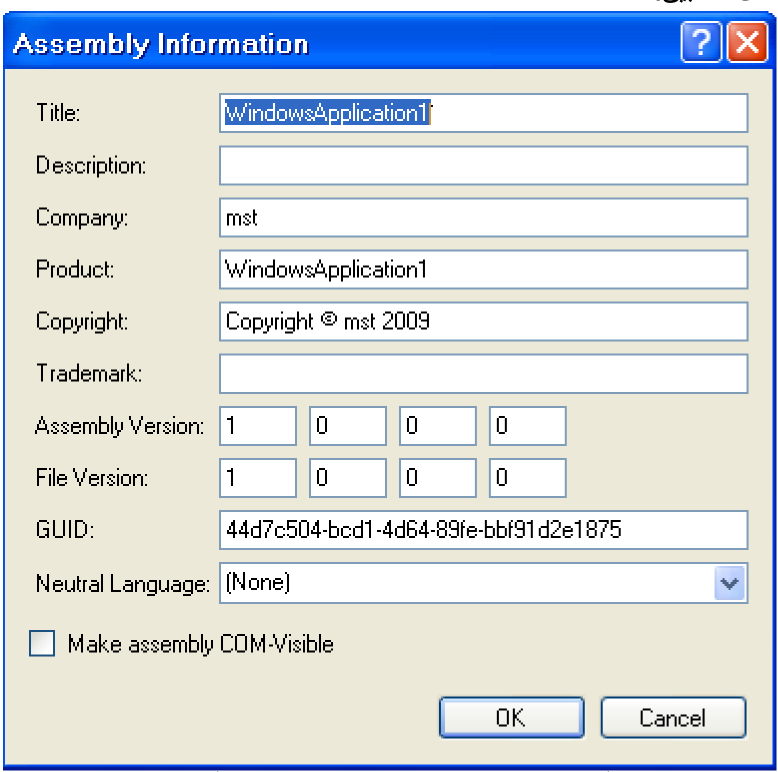
يستخدم هذا الكود العبارة Withلتخفيض كمية الكتابة المطلوبة في عبارة الإسناد الرئيسية.داخل عبارة With...End With،ليس هناك حاجة لإعادة كتابة اسم الكائن الذي يظهر تماماً بعد الكلمة المحجوزة With،في هذه الحالة My.Application.Info.Version.بإمكانك الإشارة لأعضاء ذلك الكائن بكتابة النقطة فقط(.)متبوعة باسم العضو.

وضع رقم النسخة

إذا شغلت البرنامج،فإنه سيعرض رقم النسخة المحددة حالياً,"1.0Revision 0" كما هو مبين في الشكل التالي:

مشـروع المكتبة
version1.0revision0

سؤالي،هو"أين يتم تحديد رقم النسخة هذا،وكيف استطيع تغيره؟"يتم تخزين قيم الإصدار كميتاداتا(توصيف بيانات metadata) ضمن المجمع.تتضمن الفيجوال أستوديو نموذج يسمح لك تعديل الميتاداتا المعلوماتي الأساسي المخزن في المجمع.للوصول للفورم،اعرض خصائص المشروع(أنقر مزدوج على My Project في مستكشف الحلول)،واختر تبويب Application،ومن ثم انقر على زر Assembly Information (شاهد الشكل التالي)يستخدم نموذج تطبيقنا رقم النسخة ،والذي تم وضعه باستخدام أربع حقول نصية بجانب ملصقة إصدار المجمع.وهذه الحقول الأربع تمثل أرقام المجمع Major، Minor، Build، وRevision.بإمكانك تغيرها ومن ثم نقر موافق OK ومن ثم شغل التطبيق.



على الرغم من أن هذه النموذج مريح،فهو عبارة عن مثال أخر لكتابة بعض كود مشروعك بالفيجوال أستوديو بالنيابة عنك.كل حقل على هذا النموذج يتم حفظه في ملف الكود المصدري المضمن في مشروعك.لعرضه،تأكد من أن الزر "أظهر جميع الملفات Show All Files" مازال مختار في مستكشف الحلول Solution Explorer.مدد بند My Project باستخدام إشارة الزائد،ومن ثم انقر مزدوج على بند *AssemblyInfo.vb*. هذا الملف يعرف عدة مواصفات خاصة بالمجمع assembly-specific attributes (وسنستكشفها في الفصل 18)،إنه يتضمن المدخلات العملية التالية:

```
Imports System
Imports System.Reflection
Imports System.Runtime.InteropServices
' General Information about an assembly is controlled through the following
' set of attributes. Change these attribute values to modify the information
' associated with an assembly.
' Review the values of the assembly attributes
<Assembly: AssemblyTitle("LIBRARY")>
<Assembly: AssemblyDescription("")>
<Assembly: AssemblyCompany("mst")>
<Assembly: AssemblyProduct("LIBRARY")>
<Assembly: AssemblyCopyright("Copyright © mst 2009")>
<Assembly: AssemblyTrademark("")>
<Assembly: ComVisible(False)>
'The following GUID is for the ID of the typelib if this project is exposed to COM
<Assembly: Guid("44d7c504-bcd1-4d64-89fe-bbf91d2e1875")>
' Version information for an assembly consists of the following four values:
'       Major Version
'       Minor Version
'       Build Number
'       Revision
' You can specify all the values or you can default the Build and Revision Numbers
' by using the '*' as shown below:
' <Assembly: AssemblyVersion("1.0.*")>
<Assembly: AssemblyVersion("1.0.0.0")>
<Assembly: AssemblyFileVersion("1.0.0.0")>
```

كما ترى،هذا الملف تم تحديثه بواسطة القيم التي كتبتها ضمن نموذج معلومات المجمع.هل تشاهد مواصفة AssemblyVersionالمعرفة هنا.إذا عدلت هذه القيم،فإن التغيرات ستنعكس في نموذج معلومات المجمع،وأيضاً في تطبيقك المشغل وأخيراً في المجمع المترجم.إحفظ العمل من (ملف File>> حفظ الجميع Save All)

إضافة الفورم الرئيسي Adding the Main Form

بقدر الفائدة والمواصفات الكثيرة في نموذج AboutProgramالذي عملناه حتى الآن فإن مثل هذا النموذج نادراً ما يكون عليه التركيز الجوهري لتطبيق ما.في مشروع المكتبة ،سيتم عرض هذه الفورم فقط عندما يتم طلبها من الفورم(النموذج)الرئيسيMain ،لذلك لنضيف فورم رئيسي بسيط الآن.في الفيجوال أستوديو،أختر مشروع Project >>إضافة نموذج ويندوز Add Windows Form.عندما يظهر حوار إضافة بند جديدAdd New Item، أختر نموذج ويندوز Windows Form من بنود القائمة المتاحة،وسمه MainForm.vb قبل النقر على زر إضافة Add.

من نافذة الخصائص خصص الخصائص التالية كما هو مبين في الجدول التالي:

الخاصية	القيمة التي سيتم إسنادها
(Name)	MainForm
FormBorderStyle	FixedSingle
MaximizeBox	False
Size	576, 459
Text	مشروع المكتبة

من صندوق الأدوات أضف أداة زر للفورم بالخصائص التالية:

الخاصية	القيمة التي سيتم إسنادها
(Name)	ActHelpAbout
Size	80, 24
Text	&حول...

الحرف الخاص& يعمل على وضع اختصار shortcut للزر.عندما تضغط على مفتاح Altمع الحرف الذي يتبع"&" (في هذه الحالة حرف ح)فإن البرنامج سيتصرف وكأنك نقرت على الزر بواسطة الماوس.

انقر مزدوج على الزر وأضف الكود التالي لإجراء حدث النقر:

```
Private Sub ActHelpAbout_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ActHelpAbout.Click
        ' ----- Show the About form.
        AboutProgram.ShowDialog()
    End Sub
```

هنا خصصنا مؤشر مباشر للنموذج AboutProgram.قبل الإصدار2005 من الفيجوال بيسك،كان إظهار الفورم يتطلب إنشاء حالة(نسخة) من الفورم قبل إظهارها:

```
(New AboutProgram).ShowDialog( )
```

هذا التركيب مازال يعمل،وهي الطريقة المعتمدة إذا كنت تريد عرض عدة نسخ من نفس الفورم على الشاشة في نفس الوقت.مهما يكن التركيب AboutProgram.ShowDialog( )أوضح بكثير للاستخدام المفرد للنماذج،ويعكس عن قرب كيفية تقديم الفورم .عملياً،هذه العبارة تستخدم فضاء الأسماء My.العبارة الكاملة تبدو مشابهة للتالي:

```
My.Forms.AboutProgram.ShowDialog()
```

يسمح لك التجمع My.Forms من الإشارة إلى أي فورم ضمنه دون أن يكون عليك الكتابة في البدايةMy.Forms .أعضاء التجمع تمثل الحالات الافتراضية *default instances* لكل نموذج في المشروع.

هذا كل الكود الذي نحتاجه حالياً،ولكن إذا شغلت التطبيق ،فإنه مايزال يعرض فقط الفورم AboutProgram.هذا لأن الفورم AboutProgram قد تم وضعها كفورم بداية التشغيل"startup".لتبديل هذا ، افتح نافذة خصائص المشروع project's properties،واختر تبويب Application تطبيق وضع حقل"فورم بداية التشغيلStartup form " إلى " MainForm "

سيتم الآن عرض الفورم AboutProgramكفورم"حوار"dialog" (من خلال استدعاءها بالطريقة ShowDialog)كل فورم تتضمن الخاصية DialogResultوالتي يتم إرجاع قيمتها بواسطة الطريقة ShowDialog عندما تغلق الفورم. كل زر buttonعلى نموذجك يمكن تركيبه لإسناد هذه الخاصية بشكل آلي ويغلق الفورم.فالزر "إغلاق" على الفورم AboutProgram يعمل هذا تماماً،فخاصيته DialogResult تم إسنادها إلى "إلغاء Cancel "،والتي تم إسنادها إلى خاصية الفورم DialogResult عندما ينقر المستخدم على زر "إغلاق".كتأثير جانبي،في أي وقت ، القيمة(مالم تكن لاشيء)يتم إسنادها إلى خاصية DialogResultالنموذج، وتغلق الفورم.

الحصيلة upshot من هذه الفقرة الطويلة هي أنك تستطيع الآن حذف معالج حدث النقر على زر الإغلاق،وسيبقى الزر يغلق الفورم.احذف الإجراء ActClose_Click من الكود المصدري للنموذجAboutProgram ، شغل البرنامج ولاحظ ما يحصل.الزر إغلاق مازال يعمل على إغلاق النموذج AboutProgram،حتى بدون معالج الحدث.

بإمكانك أيضاً ترك الإجراء في مكانه ،واعمل على تنظيف خاصية DialogResult للزر "إغلاق" وأضف العبارة التالية لمعالج حدث نقر الزر:

```
Me.DialogResult = Windows.Forms.DialogResult.Cancel
```

هذه ثلاث طرق مختلفة نستطيع بها إغلاق الفورم AboutProgram.يوجد عدة طرق أخرى مختلفة لإتمام نفس المهمة،لذلك كن مبدعاً.

اعتماد إضافي:إضافة أيقونة Extra Credit: Adding an Icon

لإضافة أيقونة مخصصة للفورم الرئيسي.اتبع الخطوات التالية:

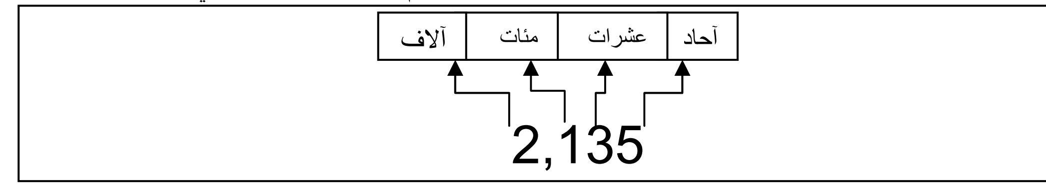
1.أعرض الفورم الرئيسي بالنقر المزدوج على البند MainForm.vb في مستكشف الحلولSolution Explorer .
2.اختر سطح الفورم
3.اختر خاصية Iconللفورم في نافذة الخصائص Properties.
4.انقر الزر "..." في هذه الخاصية ،وابحث عن ملف في الدليل الفرعي للفصل الخامس،بإمكانك أيضاً استخدام أي ملف أيقونات *ico*. .

البيانات وأنواعها Data and Data Types

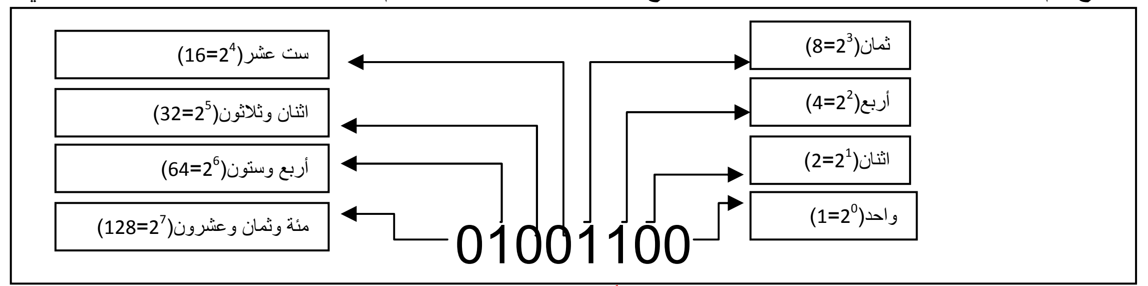
سنناقش كيف تستخدم وتعالج الفيجوال بيسك البيانات ضمن تطبيقاتك،وكيف تستطيع تسيد الأدوات التي تجعل هذه المعالجة ممكنة.

طبيعة بيانات الكمبيوتر The Nature of Computer Data

في الفصل الثاني ذكرت كيف تتحلل جميع البيانات في الكمبيوتر ضمن بتاتات(وحدات bits)مستقلة،النبضات الكهربائية التي تمثل إما 1 أو 0، on أو off،صواب true أو خطأ false. بما أن نظام الأرقام العشرية يتطلب أكثر من هذه القيم فقط،فإن الكمبيوتر يعمل في عالم من الثنائيات-نظام الأعداد المحدودة لرقمين 0 و 1.
لحسن الحظ،من السهل جداً تمثيل الأعداد العشرية الصحيحة الأساسية باستخدام مجموعة رموز الثنائي.



لاحظ الشكل التالي الذي يبين قيم الأماكن المختلفة للأعداد المتعددة الأرقام،نفس النوع من التخطيط يمكن أن يتم استخدامه من أجل الأعداد الثنائية ،فقط أسماء وقيم الموضع يتم تغيرها.من أجل الراحة،ندعو هذه المواضع بأسمائها العشرية،أو نستخدم ما يتعلق بالأساس 2.وكل هذا مبين في الشكل التالي:



موضع"8بت"8أرقام لعدد ثنائي.

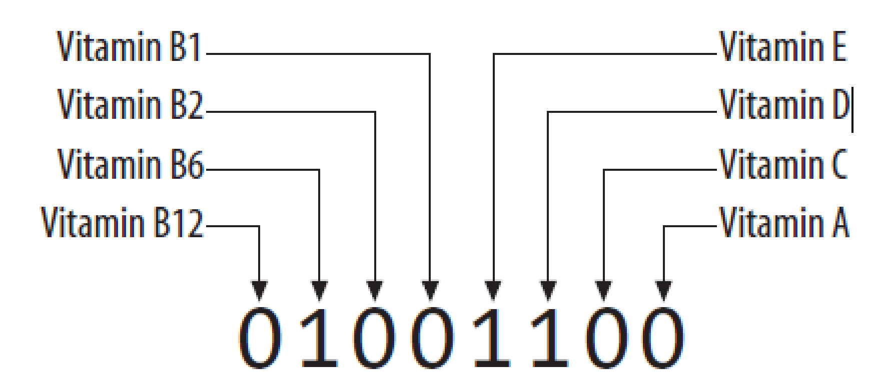
لاستكشاف ما هو الرقم المقابل في النظام العشري،فقط اجمع الأعمدة،لنرى ذلك،يوجد واحد في الموضع الثاني والثالث،وفي الموضع السادس أيضاً،ولاشيء(صفر في باقي المواضع كما هو موضح في الرقم الثنائي01001100 ولحساب قيمة كل موضع كما هو مبين في الشكل السابق نرفع الموضع إلى قوة(أس) الموضع بالنسبة للأساس2)ومن ثم نضرب كل قيمة لهذا الموضع بالقيمة الموافقة في العدد الثنائي ومن ثم نجمع النتيجة لنحصل على الرقم العشري الموافق كما يلي:

$1\times0+2\times0+4\times1+8\times1+16\times0+32\times0+64\times1+128\times0=76$ وهو العدد العشري المقابل للعدد الثنائي.

بما أن أي رقم ثنائي لايمكن أن يكون أكثر من 1،فالعد بسيط جداً.لقد بينت مثال عن 8-بت(8-رقم)ثنائي هنا-والذي بدوره بإمكانه معاملة الأعداد العشرية من0 حتى255 ولكن إمكانك تمثيل أرقام عشرية أكبر بإضافة أرقام ثنائية أكثر.ولكن هذا مناسب من أجل القيم العددية الصحيحة،فكيف ستمثل الأعداد العشرية والكسرية؟ وماذا حول الأعداد السالبة،في أي مكان ستكون مناسبة في هذا النظام الثنائي؟فهي ليست مجرد أعداد.فحاسبي بإمكانه معالجة البيانات النصية،مصفوفات الأعداد،الصور الرسومية،وسجلات مخصصة.فكيف تم تخزين مثل تلك الأشياء في النموذج الثنائي؟لمعالجة آلاف من النماذج البيانية،كل كمبيوتر يتضمن مجموعة صغيرة من الوحدات المتناهية في الصغر والتي تكون مناسبة بالنسبة للرياضيات،اللغة،والفن.

تنفذ الكمبيوترات "أنواع البيانات *data types*" المعالجة جميع النماذج المتنوعة من البيانات لكي تكون قابلة للإدارة.يتصرف كل نوع من أنواع البيانات وكمفسر (مترجم interpreter) بين تجمع من البتات وقطعة من المعلومات التي يستخدمها لكمبيوتر بحيث يستطيع مستخدم الكمبيوتر من فهمه والانتفاع utilize منه أفضل.

أخيراً ultimately تخزن أنواع البيانات محتوياتها كبتات مستقلة من البيانات،ولكنها تختلف في كيف يتم ترجمة(تفسير) هذه البتات.تصور نوع بيانات اسمه فيتامين Vitamin والذي يحدد أي فيتامين تم تضمينه في منتج طعام.يبين الشكل التالي كيف يمكن إسناد 8-بت المستخدم سابقاً وتفسيره(ترجمته) كالفيتامينات.
مع وجود نوع بيانات،بإمكانك إسناد قيمة فيتامين لبند طعام موجود في تطبيقك.(مجرد مثال)



خذ العدد 76 الذي ناقشناه سابقاً،فهو سهل التحويل جداً إلى التمثيل الثنائي،كما في 01001100.يتضمن إطار عمل الدوت نت عدة أنواع بيانات تقوم بمثل هذا التحويل بشكل آلي،تختلف فقط بالعدد المكون من أرقام ثنائية(بتات) التي يمكنها معالجتها،في عالم الكمبيوتر ،يمثل 76 أيضاً حرف من الأبجدية-الحرف الكبير *L*.وهذا لأن هناك نوع بيانات يؤسس قاموس بين القيم الثنائية والحروف الأبجدية(والرموز الأخرى). تستخدم برامج ويندوز ولمدة طويلة الآسكي *ASCII* (الكود الأمريكي القياسي لتبادل المعلومات American Standard Code for Information Interchange)وأعدادها إلى وبالعكس كقاموس الحروف.فكيف تحول أنظمة المستندات 8-بت هذه الأعداد من 0 إلى255 إلى كل الحروف

المتنوعة المستخدمة في اللغة الإنكليزية،ومن ضمنها علامات الترقيم punctuationوالرموز(الحروف)الأخرى المتنوعة miscellaneous characters.
قاموس أخر.يونيكود *Unicode* يستخدم 16-بت من البيانات لمعاملة حوالي65,000حرف مختلف.تستخدم الدوت نت اليونيكود من أجل حروفها وأنواع بيانات "السلاسل النصيةstring ".
يوجد نوع بيانات أخر وهو المنطقيةBoolean ،والتي تستخدم بت مفرد single bit لتمثيل إما صواب True (بت ذو قيمة 1)أو خطأ False (0).القيم العددية الصحيحة السالبةNegative integers ،والعشرية ذات النقطة العائمة floating-pointوالثابتةfixed-point،والتواريخ dates والوقت timesوالتي تدور خارج أنواع البيانات القاعدية على الأغلب تدار بواسطة الحاسبات وتطبيقاتها.يمكن أن يتم بناء بيانات أكثر تعقيداً من هذه الأنواع القاعدية.

البيانات في الدوت نت Data in .NET

يتم تنفيذ جميع البيانات في الدوت نت كفئات classesضمن فضاء أسماء النظام System.أحد أنواع البيانات هذه هو System.Byte،والذي ينفذ قيمة عددية صحيحة 8-بت،تماماً كما ناقشناه سابقاً.فهو يحفظ قيم عددية صحيحة من 0إلى255.وهذه القيم يتم تخزينها دائماً باستخدام8بت من البيانات الثنائية،ولكنها تظهر بطريقة سحرية في النموذج العشري decimal form عندما تريد إحضارها.
يتضمن إطار عمل الدوت نت 15 من أنواع البيانات التفسيرية الجوهريةcore interpretive data types :8أنواع للبيانات العددية الصحيحة integers،3أنواع للأعداد العشريةdecimal numbers ،2نوع للبيانات الحرفيةcharacter data ،نوع بيانات موحد للتواريخ والوقت dates and times،وأخيراً نوع البيانات المنطقيةBoolean .

أنواع البيانات العددية الصحيحة Integer Data Types

تعتمد على عدد أنواع البيانات المتاحة(8من بين الأنواع الأساسية الـ15)،يمكن أن تعتقد أن معظم المبرمجين يعملون طوال اليوم مع الأعداد الصحيحة،ويمكن أن تكون محق.سواء كانت بيانات المستخدم الفعلية أو عداد حلقة أو كود مرحلي أو طرق تخزين لأنواع البيانات المعدودة(المحصية)،تظهر البيانات العددية الصحيحة في كل مكان ضمن كود الدوت نت.يعتمد مجال القيم بالنسبة لنوع البيانات العددية الصحيحة مباشرة على العدد من الأرقام الثنائية المدارة بواسطة نوع البيانات،كلما كانت الأرقام أكثر ،كان المجال أكبر.وأيضاً النصف من أنواع البيانات العددية الصحيحة تخزن كلاً من القيم الموجبة والسالبة(تدعى الأعداد الصحيحة ذات إشارة ”signed“ integers)،بينما النصف الباقي يدعم فقط الأعداد الموجبة(بدون إشارةunsigned)تجدول القائمة التالية أنواع البيانات العددية الصحيحة المضمنة في الدوت نت،وما يصاحبها من مجالات.

نوع بيانات الدوت نت.NET data	البت Bits	النمط Style	مجال القيم Range of values
System.Byte	8	Unsigned	0 to 255
System.SByte	8	Signed	–128 to 127
System.Int16	16	Signed	–32,768 to 32,767
System.UInt16	16	Unsigned	0 to 65,535
System.Int32	32	Signed	–2,147,483,648 to 2,147,483,647
System.UInt32	32	Unsigned	0 to 4,294,967,295
System.Int64	64	Signed	–9,223,372,036,854,775,808 to9,223,372,036,854,775,807
System.UInt64	64	Unsigned	0 to 18,446,744,073,709,551,615

أنظر إلى هذه الأنواع بطريقة أخرى،يبين الجدول التالي العلاقة بين الأنواع وعدد البتات التابع لكل منها ونمط المجال:

	8-bits	16-bits	32-bits	64-bits
Signed	SByte	Int16	Int32	Int64
Unsigned	Byte	UInt16	UInt32	UInt64

أنواع البيانات العشرية Decimal Data Types

عندما جاءت الكسورfractions ، لم تكن في البداية سيئة،بما أن العديد منها يمكن أن يتم تحويله إلى نموذج عددي بسيط بإدخال الفاصلة(النقطة) العشرية في العدد:2/1يصبح0.5، و1/4 يصبح أطول ولكن يبقى أصغر 0.25، و1/3 يصبح ... 0.33333 ماالذي يحدث هنا؟لا أستطيع كتابة كل هذه 3.وإلا سيصبح الكتاب 2,000 صفحة أو أكثر.أخيراً،اكتشف الناس أنه في عدة حالات،فإن كتابة كل 3 لايستحق العناء،لذلك فإنهم توقفوا عند بعض النقاط.كما في 0.33333333.
هذا ليس بالدقة الكاملة،ولكنه جيد بما فيه الكفاية،وهذا هو الحال بالنسبة للقيم العشرية المعتمدة في الكمبيوتر computer-based decimal values.يمكن أن تمتلك دقة تامة حتى نقطة معينة.بعد ذلك،يقع عليك اختيار ما هو جيد بشكل كافي بالنسبة لك.يتضمن إطار عمل الدوت نت ثلاث أنواع بيانات عشرية. اثنان منهما يقبلان دقة محددة بدلاً من المجال الكبير من القيم.والثالث لديه دقة تامة(مثالية perfect accuracy)ولكن مجاله أكثر تحديداً.يبين الجدول التالي هذه الأنواع الثلاث.

نوع بيانات الدوت نت NET data type	الدقة Accuracy	المجالRange	وصف Description
System.Decimal	كاملة(Perfect)	محدود	يدعم تقريباً حوالي28 أرقام موحدة على كلا جانبي النقطة العشرية.ويمكن أن يحذف بعد هذا الموضع الرقم المتاح الأخير.فدقته ضمن هذه الأرقام.وبسبب هذا،فإنه مناسب للعمل مع العملة.كلما كانت الأرقام التي لديك أكثر على يسار الفاصلة،كلما قلت لديك الأرقام المتاحة على يمين الفاصلة العشرية،والعكس بالعكس vice versa،بالنسبة للأعداد التي ليس لديها جزء عشري يكون المجال –79,228,162,514,264,337,593,543,950,335 to 79,228,162,514,264,337,593,543,950,335. (That's 29 digits) بالنسبة للأعداد الصفرية فقط(0)على يسار الفاصلة العشرية،المجال يكون : –0.0000000000000000000000000001 to 0.0000000000000000000000000001
System.Single	غير كاملة	كبير	يوفر مجال أكبر من السابق ولكن لديه بعض المشاكل في الدقة.بعض الأحيان عندما تقوم بحساب معقد وأنت تعرف أن النتيجة ستكون صفر،فإن النتيجة

			يمكن أن تكون 0.000000000023.هي قريبة للصفر،ولكنها ليست صفر بالضبط،بإمكانك استخدام أعداد كبيرة جداً أو صغيرة جداً،للقيم السالبة،المجال هو :من3.402823E+38– إلى 1.401298E-45–.أما من أجل القيم الموجبة،فمجاله من1.401298E-45 إلى 3.402823E+38
System.Double	غيركاملة	ضخم	يشبه سابقه،ولكن بمجال أكبر.من أجل القيم السالبة،المجال هو:من 1.79769313486231E+308– إلى:4.94065645841247E-324– ومن أجل القيم الموجبة،المجال هو:من 4.94065645841247E-324 إلى: 1.79769313486232E+308

أنواع البيانات الحرفية Character Data Types

يتضمن إطار العمل نوعين من البيانات المتعلقة بالنصوص، System.Charو System.String.نوع البيانات الحرفية Charيحفظ حرف مفرد،لا أقل ولا أكثر.ففي 16بت،يحفظ حرف من أي من آلاف حروف اليونيكود.

ونوع بيانات السلاسل الحرفية String تسمح حتى حوالي 2بليون حرف من حروف يونيكود ،وأن يتم صفها مع بعضها البعض في مقطع نصي طويل.السلاسل الحرفية في الدوت نت ثابتة(لايمكن تغيرها immutable)،فحالما تعمل على إنشاء سلسلة نصية ما،فلايمكن تغيرها بأي طريقة.إذا أردت إضافة نص إلى سلسلة حرفية موجودة،فإن الدوت نت سيعمل على إنشاء وسمة(علامة)سلسلة حرفية جديدة مبنية من سلسلتين حرفيتين ثابتتين أصليتين.

على الرغم من أن الحرفية والسلسلة الحرفية أنواع بيانات مختلفة،بإمكانك بسهولة نقل البيانات بينهما،بما أنهما معتمدان على حروف اليونيكود الأساسية.

نوع بيانات الوقت والتاريخ Date and Time Data Type

يسمح لك نوع البيانات System.DateTime من تخزين إما قيم تاريخ أو وقت(أو كلاهما)كبيانات.داخلياً،إن DateTime هو عبارة عن عداد عددي صحيح بسيط يعرض تنسيق تاريخ أو وقت محول عند الحاجة.وكعدد،فإنه يحسب عدد "الدقات ticks "منذ12:00صباحاً 1. كانون الثاني ،سنة1بعد الميلاد(12:00 a.m. on January 1, 1 AD).كل دقة"tick" هي تماماً100نانوثانية،لذلك فهو دقيق جداً.التاريخ الأعظمي المسموح به هو 31كانون الأول،9999،في التقويم الغريغوري(الكاثوليكي).

نوع البيانات المنطقية Boolean Data Type

يمثل نوع البيانات System.Boolean ماهيةessence بيانات الكمبيوتر الحقيقية:وهو البتbit .فهو يحفظ واحد من قيمتين ممكنتين:صواب True أو خطأ False .

من المجحف Shockingly أن يستخدم نوع البيانات الذي يحتاج عملياً مابين 2 و4بايت(32بت)من مساحة(فراغ)بيانات لحفظ مسار بت مفرد من البيانات. مما أظهر أهمية القيم المنطقية في البرامج.وكمطور،فإنك دائماً تختبر شروط متنوعة لترى أي منها يتحقق قبل أن تعالج مقطع من كود.كل هذه الشروط تختصر boil down أخيراً لقيم منطقية Booleanوعملياتoperations . حتى أن للدوت نت طرق لنقل البيانات بسهولة بين القيم العددية الصحيحة ونوع البيانات المنطقية.في مثل هذا التحويلات ،يصبح الصفر(0)خطأً وتصبح باقي جميع القيم الممكنة صواب .وعند الانتقال من المنطقي إلى ما يكافئه من القيم العددية الصحيحة فإن خطأ يصبح(0) وصواب يصبح(-1).(إذا استخدمت لغة سي شارب #C،ستجد أنها تحول صواب إلى 1 وليس-1،داخلياً في الدوت نت، يتم تحويل صواب إلى1،ولكن لأسباب تاريخية،تستخدم الفيجوال بيسك-1.وهذا الاختلاف عادة ليس بالمشكلة مالم تخزن القيم المنطقية كأعداد صحيحة في ملف على قرص ما وتتوقع من برامج فيجوال بيسك وسي شارب #Cأن تفسر interpretالبيانات بشكل صحيح).

فئة الكائن The System.Object Class

لقد علمت مسبقاً أن الدوت نت بيئة تطوير كائنيه التوجه(موجهة بالكائناتobject-oriented development environment)،كل شيء في الدوت نت- كل الكود وكل البيانات- يتم اشتقاقها (مستمدة)من فئة قاعدية واحدة: System.Object.بنفسها هذه الفئة ليس لديها الكثير من الميزات.يمكن أن تخبرك عن اسمها،نوعها،وفيما إذا حالتين(نسختين) من كائن هما في الواقع كائن واحد ونفس الكائن.غير ذلك فهي ليست مفيدة للكثير ما عدا أنه يتم استخدامها كنقطة بداية بالنسبة لجميع الفئات والأنواع.لأن جميع الفئات في الدوت نت –ومن ضمنها كل أنواع البيانات- تستمد(تشتق)من System.Object،تستطيع معاملة حالة من أي فئة(أو نوع بيانات) ككائن .وستتذكر البيانات ما هو النوع الحقيقي الذي هي عليه،لذلك إذا كان لديك System.Int32موضوع كـ System.Object، تستطيع تغيره إلى System.Int32فيما بعد.

الأنواع ذات القيمة والأنواع المرجعية Value Types and Reference Types

ارجع إلى الفصل الأول،لقد قرأت الاختلاف بين الأنواع ذات القيمة والأنواع المرجعية.فالأنواع ذات القيمة هي خانات(حاويات buckets) تحتوي على البيانات الحقيقية، والأنواع المرجعية تحتوي تعليمات instructionsعن المكان الذي تستطيع إيجاد البيانات الحقيقية.بشكل عام،تحتوي الأنواع ذات القيمة قيم بيانات صغيرة وبسيطة،بينما الأنواع المرجعية تشير إلى مقاطع بيانات معقدة وكبيرة.وهذا ليس دائما صحيح،ولكن من أجل أغلب البيانات التي تعمل معها،فإنه سيكون صحيحاً.إن System.Object هو نوع مرجعي والذي منه يتم اشتقاق جميع الأنواع والفئات الأخرى.وهذا يتضمن جميع أنواع البيانات الأساسية،لذلك ربما تفكر أنها جميعها أنواع مرجعية أيضاً. ولكن يوجد فئة أخرى معلقة بين System.Object ومعظم أنواع بيانات الفيجوال بيسك.وهذه الفئةSystem.ValueType ، ،تنفذ التعريف الأساسي والاستخدام لنوع ما ذو قيمة.يبين الجدول التالي بعض الاختلافات بين الأنواع المرجعية وذات القيمة.

الأنواع ذات القيمة:Value types

1.تشتق في النهاية من System.ValueType،والذي بدوره يشتق من System.Object.

2.أنواع البيانات الأساسية المشتقة:المنطقيةBoolean، البايتByte، الحرفيChar، التاريخ والوقتDateTime، العشريDecimal، المزدوجDouble،القصيرInt16، الانتغرInt32، الطويلInt64، البايت بإشارةSByte، المفردSingle، القصير بدون إشارةUInt16، الانتغر بدون إشارة UInt32،الطويل بدون إشارةUInt64 .

3.توفر دعم لتراكيب الفيجوال بيسك structures.

4.يتم اشتقاق العدادات Enumerationsكما يلي:System.Enum>> System.ValueType >> System.Object.

5.لايمكن اشتقاق الأنواع ذات القيمة من فئات أخرى أو تراكيب ولا حتى يمكن للتراكيب أن تشتق منها.

6.لايمكن وضع الحالات(النسخ)إلى "لاشيءNothing "(استخدام النوع "قابل لأن يكون بدون قيمة nullable "يتغلب على هذا القصور limitation)

7.يمكن للحالات أن تحتوي فقط بيانات من نوع محدد.على سبيل المثال،نسخ(حالات) System.Int32يمكن أن تحتوي فقط بيانات عددية صحيحة integerبإشارة32-bit.

8.التبديد بالكامل يتم بعمليات مجمع المخلفات.NET garbage collection..

لأنواع المرجعية Reference types

1.في النهاية ،تشتق من System.Object.
2.أنواع البيانات الأساسية المشتقة:السلاسل الحرفية String.
3.توفر دعم لفئات الفيجوال بيسك Visual Basic "classes".
4.تستخدم التفاويض Delegates كمراجع(مؤشرات)لطرق الفئة class methods، وتشتق كما يلي:System.Delegate <<
System.Object،أحد أنواع التفاويض "multicast delegate" " يتم اشتقاقه من خلال:System.MulticastDelegate .
5.يمكن أن يتم اشتقاق الأنواع المرجعية من فئات أخرى،ويمكن أن تستخدم كفئات قاعدية (يمكن أن يتم الاشتقاق منها أيضا)
6.يمكن وضع الحالات(النسخ)إلى "لاشيء Nothing "
7.تشير عادة الحالات(النسخ) لبيانات النوع الذي عرفها(المعرفة منه)،ولكن يمكن لحالة أن تشير أيضاً إلى النوع المشتق.على سبيل المثال،حالة من System.String يمكن أن تشير لأي بيانات تستخدم System.String كفئة قاعدية base class.
8.يتم تدميرها من خلال مجمع النفايات garbage collection.

(بالإضافة للفئات والتراكيب،تعرف الفيجوال بيسك أيضاً "الوحدات البرمجية modules ".توثيق الدوت نت يحدد الوحدات البرمجية modules على أنها أنواع مرجعية،ولكنك لاتستطيع إنشاء حالات(نسخinstances)منها.)

يمكن للأنواع ذات القيمة أن تحتوي فقط بيانات من نوعها الخاص ،ولكن الأنواع المرجعية يمكن أن تشير إلى الحالات المشتقة.وهذا هام في الدوت نت،حيث أنه تم تصميمها لتسمح لحالة System.Object من الإشارة إلى أي بيانات في تطبيق ما.نسخ(حالات) System.Object يمكن أن تشير إلى إما لبيانات لنوع ذو قيمة أو لنوع مرجعي.هذا سهل الفهم بالنسبة للأنواع المرجعية بما أن تلك الحالة(النسخة)ستشير فقط لبعض الحالات المشتقة من نفسها.ولكن إذا أسندت نوع ذو قيمة إلى System.Objec المرجعي ،فعلى الدوت نت أن يعلّم(يضع علامة mark)تلك الحالة بطريقة خاصة ليشير إلى أن النوع المرجعي يحتوي نوع ذو قيمة. وهذه المعالجة(العملية) تدعى (الصندقة *boxing*)وعكس العملية يطلق عليه(*unboxing*.).على الرغم من أن الصندقة boxing مفيدة،وفي بعض الأحيان أساسية،فإنها تأتي بأداء فعلي يزعج بشدة substantial performance hit.

أنواع بيانات الفيجوال بيسك Visual Basic Data Types

جميع أنواع البيانات المنفذة في لغة الفيجوال بيسك هي أغلفة لأنواع بيانات الدوت نت الجوهرية.فقد تم تغيير فقط بعض الأسماء للمحافظة على البساطة. يبين الجدول أنواع بيانات الفيجوال بيسك وما يكافئها في الدوت نت .

Visual Basic type	.NET type
Boolean	System.Boolean
Byte	System.Byte
Char	System.Char
Date	System.DateTime
Decimal	System.Decimal
Double	System.Double
Integer	System.Int32
Long	System.Int64
Object	System.Object
SByte	System.SByte
Short	System.Int16
Single	System.Single
String	System.String
UInteger	System.UInt32
ULong	System.UInt64
UShort	System.UInt16

كل أنواع بيانات الفيجوال بيسك قابلة للتبادل بالكامل مع ما يكافئها في الدوت نت.وأي حالة من System.Int32يمكن أن يتم معاملتها وكأنها حالة من Integer،والعكس بالعكس.

المحارف Literals

الطريقة الأسرع في تضمين قيم لنوع بيانات معين في كود الفيجوال بيسك هي استخدام المحارف(الحرفية).ولقد رأيت سابقاً في الفصل الأول من هذا الكتاب كيفية استخدام الحرفية في مثال المشروع وكما يلي:

```
MsgBox("Hello, World!")
```

هذا الاستدعاء لدالة MsgBoxيتضمن محارف السلسلة الحرفية.محارف السلسلة الحرفية تظهر دائماً ضمن علامتي اقتباس مضاعفة double quotes . ومعظم المحارف العددية تظهر مع الحرف المعرف لنوع البيانات data-type-defining character في نهاية المحرفliteral .ولكن يوجد تنوع آخر،يبين الجدول التالي قائمة بقيم المحارف المختلفة التي بإمكانك تضمينها في كودك.

نوع المحرف Literal type	مثال Example	الوصفDescription
Boolean	True	يدعم نوع البيانات المنطقية قيمتين حرفيتين:صواب Trueوخطأ False.
Char	"Q"c	محارف حرفية مفردة تظهر ضمن علامتي اقتباس مضاعفة مع التذيل بالحرف c.المحارف من النوع Charليست نفس المحارف الحرفية المفردة من نوع السلاسل الحرفية String.
Date	#11/7/2005#	تظهر المحارف التاريخية والوقت بين مجموعة من العلامات العددية#.بإمكانك تضمين التواريخ، الأوقات،أو دمج منهما.يمكن لقيم التواريخ والوقت أن تكون بأي تنسيق مميز بواسطة نظام التشغيل ويندوز،على الرغم من أن الفيجوال أستوديو يمكن أن تعيد تنسيق محرف التاريخ وذلك للتوافق مع التنسيقات القياسية الخاصة بها.
Decimal	123.45D,123.45@	القيم ذات النقطة العائمة Floating-point من النوع العشري Decimal يتم إتباعها بـ D،أو بالرمز @.
Double	123.45R، 123.45#	القيم ذات النقطة العائمة من النوع Double يتم إتباعها بالحرف الكبير R، أو بالرمز#.وأيضاً،إذا استخدمت المحارف العددية مع الجزء العشري،ولكن بدون تذييل حرف لنوع البيانات،فإن ذلك المحرف سيتم إعطاء نوع البيانات من النوع المزدوج Double.
Hexadecimal	&HABCD	بإمكانك تضمين المحارف الست عشرية hexadecimalفي كودك ببدء القيمة بالتعاقب الحرفي"&H"، متبوعة بالأرقام الست عشرية hex digits .
Integer	123.45I،123.45%	القيم العددية الصحيحة من النوع Integer تكون متبوعة بالحرف الكبير I،أو بالرمز %. أيضاً،إذا استخدمت محارف عددية تقع ضمن مجال الـ Integer،ولكن دون تذييل حرف يبين نوع البيانات،فإن ذلك المحرف سيتم إعطاء النوع من نوع بيانات Integer.
Long	123.45L،123.45&	قيم عددية صحيحة من النوع الطويل Long،تكون متبوعة بالحرف الكبير L، أو بالرمز &.أيضاً،إذا استخدمت محارف عددية تقع ضم مجال الـLong ،ولكن دون تذييل حرف يبين نوع البيانات،فإن ذلك المحرف سيتم إعطاء النوع من نوع بيانات Long .
Octal	&O7654	بإمكانك تضمين المحارف الثمانية octalفي كودك ببدء القيمة بالتسلسل الحرفي"&O"،متبوعاً بالأرقام الثمانية octal digits.
Short	123.45S	القيم الصحيحة من النوع القصير Shortتكون متبوعة بالحرف الكبير S.
Single	123.45F،123.45!	القيم ذات النقطة العائمة من النوع المفرد Singleتكون متبوعة بالحرف الكبير F أو بالرمز !.
String	"A ""B"" C"	تظهر محارف السلاسل النصية Stringضمن مجموعة من علامات الاقتباس المضاعفة،دون أن يتبع علامة إغلاق الاقتباس أي حرف خاص.استخدم رمزي اقتباس ضمن محرف السلسلة الحرفية إذا أردت تضمين علامة اقتباس مفردة .

الثوابتConstants

المحارف جيدة،ولكن ليست دائما واضحة بما تعنيه.توفر الثوابت طريقة لإسناد أسماء ذات معنى للقيم الحرفية(المحارف)،ويتم معاملتها تماما كالقيم الحرفية(المحارف)،ولكن ذات تعريف وحيد،يمكن أن يتم استخدامها مرة بعد مرة في كودك.وكل استخدام لقيمة حرفية،حتى ولو كان لديها نفس القيمة،فهي تمثل تعريف جلي وحالة من القيمة.في الفيجوال بيسك يتم تعريف الثوابت باستخدام الكلمة المحجوزة Const.

```
Const MonthsInYear As Short = 12
```

تعريف الثابت لديه الأجزاء التالية:

الاسم *name*

في هذه الحالة اسم الثابت هو:MonthsInYear

نوع البيانات *data type*

يعرف هذا المثال ثابت من النوع القصيرShort . نوع البيانات دائماً تأتي بعد الكلمة المحجوزة As.
إذا لم تضع الكلمة المحجوزة As،فإن نوع بيانات الثابت ستكون من نوع المحارف التي تم إسنادها له.وأنواع البيانات التالية فقط يمكن أن يتم استخدامها من أجل الثوابت:المنطقيةBoolean،البايتByte،الحرفيChar،التاريخDate،العشريDecimal،المزدوجDouble،العددي الصحيحInteger،الطويلLong،الكائن Object،البايت بإشارةSByte،القصيرShort،المفردSingle،السلسلة النصيةString،العددي الصحيح بدون إشارة UInteger،الطويل بدون إشارةULong،القصير بدون إشارةUShort،أو اسم عداد ماenumeration .

الممهد *initializer*

الممهد المسند هنا هو12.حالما يتم إسنادها،فإن هذه القيمة لايمكن أن يتم تعديلها بينما يكون كودك في وضع التشغيل.والثوابت دائماً أنواع ذات قيمة،وليست من النوع المرجعي.والممهدات عادة تكون محارف بسيطة،ولكن بإمكانك دائماً تضمين حسابات بسيطة:

```
Const Seven As Integer = 3 + 4
```

مستوى الوصول *access level*

يمثل تعريف الثابت تنسيق مثالي MonthsInYearلتعريف ثابت مضمن ضمن كود إجراء.بإمكانك أيضاً تعريف ثوابت خارج إطار الإجراءات،ولكن يبقى ضمن فئة أو نوع أخر.عندما تعمل هذا،عادة تعمل على إضافة الكلمة المحجوزة لمحدد الوصول *access modifier* تماماً قبل الكلمة المحجوزة Const.وهذه الكلمة المحجوزة تشير إلى الكيفية التي بها سيستخدم معظم الكود هذا الثابت.وسأشرح محددات الوصول *access modifier* فيما بعد،ولكن الثوابت المعرفة ضمن إجراء يمكن استخدامها فقط ضمن ذلك الإجراء.حالما تعرف ثابت،بإمكانك استخدامه في أي مكان تود استخدام ما يكافئه من المحارف.

```
Const GreatGreeting As String = "Hello, World!"
...Later...
MsgBox(GreatGreeting)
```

العدادات Enumerations

العدادات واحدة من أنواع الدوت نت الجوهرية،تتيح لك تجميع مسميات،وقيم عددية صحيحة متعلقة بهذه المسميات كمجموعة.حالما تربطهما بعضهما، يمكن للعداد أن يتم استخدامه مثل أي نوع بيانات أخر،بإمكانك إنشاء متغيرات بحيث تكون حالات خاصة من عداد.
العدادات هي تشييد متعدد الأسطر،يعرف السطر الأول الاسم ونوع البيانات الملقاة تحت العداد.كل عضو عداد يظهر في سطر منفصل،يتم الإنهاء بسطر End Enum الذي في النهاية يغلق العداد.

```
Enum CarType As Integer
    Sedan = 1
    StationWagon = 2
    Truck = 3
    SUV = 4
    Other = 5
End Enum
```

يتضمن السطر الأول التصريح عن العداد بالكلمة المحجوزة Enum،واسم العداد(CarType)،ونوع البيانات المضمن(Integer).والتعبير كـ Asنوع البيانات هو اختياري،فإذا لم تضعه،فإن النوع الافتراضي للعداد هو العدد الصحيحInteger .فإذا زودت بنوع البيانات،فيجب أن يكون واحد من الأنواع التالية: البايتByte، العددي الصحيحInteger، الطويل Long،البايت بإشارةSByte، القصيرShort، العددي الصحيح بدون إشارة UInteger،الطويل بدون إشارةULong، أو القصير بدون إشارة UShort.
كل عضو من العداد(السطر من 2إلى 6)يجب أن يتضمن على الأقل اسم عضو(مثل Sedan).بإمكانك بشكل اختياري إسناد قيمة عددية لبعض أو لجميع الأعضاء،كما فعلنا بالمثال.فإذا ما نقص عضو إسناد ما،فيتم وضعه بزيادة واحد عن العضو السابق.أما إذا لم يتم إسناد أي عضو لقيمة،فيتم إسناد الأول لـ0، والتالي 1،وهكذا. حالما يتم التعريف،تتصرف أعضاء العداد بكثير من الشبه للثوابت العددية الصحيحة،بإمكانك استخدامها في أي مكان تريد استخدام عادةً الثوابت أو المحارف.عندما تشير لأعضاء عداد في كودك،ضمن كلاً من اسم العداد واسم العضو.

CarType.Sedan

العبارة Enumلايمكن استخدامها ضمن طريقة methodأو إجراءprocedure. بدلاً من ذلك،إنك تعرف عداد كعضو من نوع(فئة class،تركيب structure،أو وحدة برمجية module)، أو كنوع خاص قائم بذاته،تماماً مثل الفئة.يتضمن إطار عمل الدوت نت العديد من العدادات المفيدة المعرفة مسبقاً معدة intended للاستخدام مع ميزات features إطار العمل. على سبيل المثال،العداد System.DayOfWeek يتضمن أعضاء لكل أيام الأسبوع.

المتغيرات Variables

المحارف مفيدة وجيدة،والثوابت والعدادات أفضل،ولكن ولا واحد منهم يمكن أن يتم تبديله حالما يبدأ البرنامج.وهذا مايميل لجعل تطبيقك جامد وغير مرن. المتغيرات هي حاويات مسماة من أجل البيانات،مثل الثوابت تماماً،ولكن يمكن أن يتم تعديل محتوياتها أثناء تشغيل التطبيق.وأيضاً،يمكن أن تحتوي على أنواع ذات قيمة وأنواع مرجعية.إليك التركيب الأساسي لتعريف متغير جديد:

```
Dim customerName As String
```

الكلمة المحجوزة Dimأصلاً من الكلمة *dimension*–تعرف متغير جديد،في هذه الحالة،سمي المتغير customerNameمع نوع البيانات من نوع السلسلة الحرفية String.هذه الحاوية المسماة جاهزة لحفظ أي قيمة نصية،أسند له محارف نصيةstring literals، أو متغيرات نصية أخرى string variables،أو قيم معادة من دالات functionsتولد سلاسل حرفيةstrings. وبما أنه من النوع المرجعي،فيمكن أن يتم وضعه أيضاً إلى "لاشيء Nothing "(قيمة خاصة بالفيجوال بيسك وهي كلمة محجوزة تعني"هذا النوع المرجعي فارغ،فارغ بكل معنى الكلمة")

```
customerName = Nothing   ' Nothing,لاشيء
customerName = "محمد"  ' Literal حرفي
customerName = GetCustomerName(customerID)  ' Function result,نتيجة دالة
```

جميع المتغيرات تحتوي قيمتها الافتراضية حتى تضع لها شيء ما آخر.من أجل الأنواع المرجعية والأنواع القابلة لوضع بدون قيمة nullable،القيمة الافتراضية هي"لاشيء Nothing "، أما من أجل القيم العددية الافتراضي هو 0، المنطقية Booleansالافتراضي هو خطأ False،بإمكانك تضمين إسناد أولي كجزء من عبارة Dim لإعادة قيادة overrideالإسناد الافتراضي.

```
Dim countdownSeconds As Short = 60
Dim processingDate As Date = Today
Dim customerName As String = GetCustomerName(customerID)
```

السطر الأخير يبين نوع مرجعي-سلسلة حرفية-تم إسناد لها نتيجة سلسلة حرفية لدالة .بإمكانك أيضاً إسناد حالة موسومة بـ newلحالة مرجعية إلى متغير من النوع المرجعي.وبالتالي فهو جديد،فهو يستخدم الكلمة المحجوزة الخاصة New،والتي تقول"إني أعمل على إنشاء حالة جديدة من نوع بيانات خاص." يوجد العديد من التنوعات المختلفة ولكن كلها تعطي نفس النتيجة.

```
' ----- One-line variation.
    Dim someEmployee As New Employee
    ' ----- Another one-line variation.
    Dim someEmployee As Employee = New Employee
    ' ----- Two-line variation.
    Dim someEmployee As Employee
someEmployee = New Employee
```

تذكر أن الأنواع المرجعية هي معلبات buckets تحتوي على أدلة directionsلموقع البيانات الحقيقي،فعندما ينبثق متغير مرجعي للوجود للمرة الأولى،فإنه يحتوي على لاشيءNothing . وهذا كل شيء،فالمعلبة bucket لا تحتوي تعليمات instructionsعلى الإطلاق بما أنه لا يوجد بيانات مرتبطة مخزنة في أي مكان.وعندما تسند حالة جديدة new instance لمتغير من النوع المرجعي،فإن تلك الحالة يتم تخزينها في مكان ما في الذاكرةmemory ، وتعليمات عن موقع تلك البيانات يتم إفراغها dumpedضمن المعلبة.في مقطع الكود السابق،كل استخدام للكلمة المحجوزة Newيعمل على إنشاء حالة(نسخة)بيانات جديدة في مكان ما في الذاكرة.وموقع هذه البيانات يتم إسناده فيما بعد إلى المتغير someEmployee.
تتضمن العديد من الفئات واحد أو أكثر من المشيدات *constructors*، الإجراءات الممهدة initialization routines التي تثبت القيم الأولية initial values لحالة.بإمكانك استدعاء مشيد خاص من خلال العبارة New.يتضمن نوع بيانات السلسلة النصية string مشيدات constructorsتتيح لك بناء سلسلة نصية أولية.واحد من هذه المشيدات الخاصة يتيح لك إنشاء نص يحتوي على عدة نسخ من حروف خاصة.العبارة التالية تعمل على إسناد سلسلة حرفية من 25نجمة للمتغير lotsOfStars.

```
Dim lotsOfStars As New String("*"c, 25)
```

سيتم مناقشة المشيدات بالتفصيل في الفصل 8 إن شاء الله.

يمكن لعبارة Dimأن تظهر في أي مكان ضمن إجراء،ولكن بالنسبة للتقليد فإنها تظهر تماماً عند بداية الإجراء،قبل أي عبارة منطقية أخرى.

```
Sub MyProcedure()
        Dim myVariable As Integer
...الكود الإضافي يتم وضعه هنا    '
    End Sub
```

كما مع الثوابت،يمكن أن يتم تعريف المتغيرات إما ضمن إجراء،أو خارج إجراء ولكن ضمن نوع ما.(المتغيرات والثوابت المصرح عنها خارج إجراء تعرف بالحقول*fields*. والمتغيرات والثوابت المصرح عنها ضمن إجراء تعرف أيضاً بالمتغيرات المحلية *local variables*والثوابت المحلية *local* *constants*، على الترتيب.)يتم استبدال الكلمة المحجوزة Dimبواسطة واحد من محددات الوصول *access modifiers* فيما يخص الحقول *fields*:

Private خاص:

المتغيرات الخاصة Private variables يمكن أن يتم استخدامها بواسطة أي عضو أو إجراء ضمن النوع،ولكن ليس في أي مكان أخر.إذا عملت على اشتقاق فئة جديدة new class من فئة قاعدية base class والتي تتضمن متغير لنوع خاصprivate، فإن الكود في تلك الفئة المشتقة لن يكون له إمكانية الوصول على الإطلاق إلى ذلك المتغير الخاص ،حتى أنه لن يعلم أبداً أنه موجود.

Friend صديق

المتغيرات الصديقة هي خاصة بمجمع ماassembly. يمكن أن يتم استخدامها بواسطة أي كود في النوع المرتبطة به.ولكن أيضاً بواسطة أي كود في أي مكان في نفس المجمع.حالياً هذه هي المتغيرات الصديقة.

Public العام

المتغيرات العامة تكون متاحة في أي مكان.فهي ممكنة لكتابة تطبيق ما أو مكون يعرض أنواعه لكود أبعد من نفسه.وأي شيء يتم تعليمه كعام يمكن أن يتم عرضه في هذه الطريقة.

Protected المحمي

المتغيرات المحمية تشابه كثيراً المتغيرات الخاصة،ولكن الكود في الفئات المشتقة يمكن أيضاً أن تصل إليها.بإمكانك استخدام الكلمة المحجوزة Protected
فقط في تعريف الفئةclass ، فلن تعمل في التراكيب structureوالوحدات البرمجيةmodule .

Protected Friend الصديق المحمي

تجمع المتغيرات الصديقة والمحمية كل ميزات الصديق والمحمي.يمكن أن يتم استخدامها فقط في الفئات.
يمكن لفئة مفردة أو نوع أن يحتوي على الحقول fieldsوالمتغيرات المحلية local variables والثوابت constants.

```
Class mclass
        ' ----- Here's a field.
        Private InternalUseOnly As Boolean
        Sub MyProcedure()
            ' ----- Here's a local variable.
            Dim myVariable As Integer
        End Sub
    End Class
```

توجد تنوعات تركيبية أخرى لعبارة Dim،بعض منها سأناقشه فيما بعد في هذا الفصل والفصول الأخرى.

Scope and Lifetime العمر والمدى

عندما تعرف متغير ضمن إجراءprocedure، فإن له مدى على مستوى الإجراء *procedure-level scope*. وهذا يعني أنك تستطيع استخدام المتغير في أي مكان ضمن الإجراء.سيكون لإجرائك على الأرجح "عبارات مقطعية block statements" وهذه العبارات،مثل For...Nextو If...Then والتي تتطلب أكثر من سطر وحيد من الكود المصدري لاكتمالها. إذا أضفت عبارة Dimبين سطري بداية ونهاية واحد من هذه العبارات،فإن ذلك المتغير المصرح عنه سيكون لديه فقط مدى على مستوى المقطع*block-level scope*. وسيكون متاح فقط ضمن ذلك المقطع من الإجراء.

```
For counter = 1 To 10
                Dim processResult As Integer
                ' ----- More code here.
            Next counter
            MsgBox(processResult) ' هذا السطر من الكود سيفشل
```

يصرح هذا الكود عن processResultضمن مقطع For...Next.لذلك،فهو متاح فقط للاستخدام داخل ذلك المقطع،وأي محاولة لاستخدام خارج إطار مقطع Forفسينتج عنه خطأ مباشرimmediate error .
إن عمر *lifetime* المتغير على مستوى الإجراء procedure-level variable يبدأ عندما يدخل الكود للمرة الأولى ذلك الإجراء،وينتهي عندما يخرج الكود الإجراء. وهذا صحيح بالنسبة لكل من المتغيرات على مستوى المقطع وأيضاً على مستوى الإجراء.وهذا يعني أنك إذا أسندت لمتغير على مستوى مقطع قيمة ما قبل أن يوجد المقطع، فإنه(المتغير)سيبقى لديه تلك القيمة إذا ما عدت ودخلت ذلك المقطع خلال الاستدعاء لنفس الإجراء.
بالنسبة للحقول fields(المتغيرات على مستوى الفئةclass-level variables)،يعتمد المدى scopeعلى مستوى الوصول المستخدم عند التصريح عن المتغير.وعمر الحقل يبدأ عندما يتم إنشاء نسخة(حالة) من الفئة في الكود.وينتهي عندما يتم تدمير الحالة أو تخرج تماماً من الاستخدام .

Variable and Constant Naming Conventions تقاليد تسمية المتغيرات والثوابت

لا تعليق.

Local Type Inference استنتاج النوع المحلي

operators المعاملات

يتضمن الفيجوال بيسك تشكيلة من المعاملات التي تتيح لك معالجة قيم متغيراتك.لقد رأيت سابقاً معامل الإسناد*assignment operator* (=)،والذي يتيح لك إسناد قيمة مباشرةً لمتغير.معظم المعاملات الأخرى تتيح لك بناء تعابير تضم عدة قيم أصلية بطرق منهجية(قانونية) للإسناد النهائي لمتغير.خذ العبارة التالية:

```
squareArea = length * width
```

تتضمن هذه العبارة معاملين:الإسناد assignment والضرب multiplication.معامل الضرب يوحد قيمتين(الطول والعرض)باستخدام الضرب.ومعامل الإسناد يخزن الناتج في المتغير squareArea.بدون المعاملات، ستكون تحت ضغط كبير لحساب المساحة area أو أية صيغة أخرى قد تكون معقدة complex formula.

يوجد نوعين من المعاملات غير الإسنادية non-assignment operators: أحادي *unary* وثنائي *binary*: معاملات الأحادية تعمل فقط مع قيمة وحيدة،أو مع عامل Operands . معاملات الثنائية تتطلب عاملين،ولكن النتيجة في معالجة وحيدة. العوامل Operands تتضمن المحارف(المحرفية literals)،الثوابت constants، المتغيرات variables، وقيم الدالة لمعادة function return values. يبين الجدول التالي معاملات مع تفاصيل الاستخدام.

معامل Operator	الوصف Description
+	(الإضافة Addition).يعمل على إضافة عاملين مع بعضهما،لإنتاج مجموع sum. بعض المبرمجين يستخدم هذا المعامل لإتمام توحيد سلسلة حرفية string concatenation،ولكن من الأفضل ضم السلاسل الحرفية باستخدام معامل أخر(&)والمصمم خصيصاً لذلك الهدف. التركيب:عامل 1 *operand1*+عامل 2 *operand2*.مثال:3+2
+	الموجب الأحادي Unary plus. يضمن عودة عامل بإشارته الحالية،إما موجب أو سالب.حيث أن معاملات بشكل آلي تحفظ إشارتها،وهذا المعامل عادة زائد(فائض redundant)وقد يأتي مساعد عندما نناقش "إعادة قيادة معاملات operator overloading". التركيب:+عامل operand+.مثال:+5 .
-	الطرح. يطرح عامل(الثاني)من أخر(الأول)ويعود بالفرق.التركيب: operand2 – *operand1*.مثال:10-4.
-	الناقص الأحادي Unary negation. يعكس إشارة عامله.عندما يتم استخدامه مع الأعداد المحرفية،فإنه ينتج قيمة سالبة.وعندما يستخدم مع متغير يحتوي قيمة سالبة فإنه يعطي نتيجة موجبة. التركيب:operand2– مثال: 34- .
*	الضرب.يضرب عاملين مع بعضهما،ويعود بالناتج.التركيب: operand2 * operand1.مثال:3 * 8 .
/	القسمة.يقسم عامل(الأول)على آخر(الثاني)،ويعود بحاصل القسمة.إذا كان معامل الثاني يحتوي 0،والتقسيم على صفر ينتج عنه خطأ.(عند العمل مع قيم المفرد Single والمزدوج Double، فإن التقسيم على صفر يعود بدليل "لانهاية" أو "ليس بعدد" . التركيب: operand1 / operand2.مثال 9 / 3 .
\	القسمة الصحيحة Integer division. تقسم عامل (الأول) على عامل أخر(الثاني)وتعود بحاصل القسمة،أولاً truncating تحذف أي جزء عشري من تلك النتيجة. وكذلك إذا كان معامل الثاني يحتوي صفر،والقسمة على صفر يحدث خطأ(راجع قائمة التنبيهات مع المعامل /). التركيب: operand1 \ operand2.مثال:4 \ 9 .
Mod	الباقي Modulo. يقسم عامل (الأول)على آخر(الثاني)،ويعود بالباقي كقيمة صحيحة.إذا كان معامل الثاني يحتوي على صفر،فالتقسيم على الصفر يحدث عنه خطأ(راجع قائمة التنبيهات مع المعامل /).التركيب: operand1 Mod operand2 مثال: 10 Mod 3.
^	الأس(القوة Exponentiation).يرفع عامل (الأول) إلى قوة الأخر(الثاني).التركيب: operand1 ^ operand2.مثال: 8 ^ 2.
&	ضم النصوص(السلاسل الحرفية String concatenation).يضم عاملين لبعضهما،ويعود بنتيجة سلسلة حرفية موحدة.كلا معاملين يتم تحويلهما إلى ما يكافئهما من السلسلة الحرفية String قبل أن يتم ضمهما مع بعضهما.التركيب: operand1 & operand2.مثال: "K" & "O".
And	عملية "و"،حرف العطف.Conjunction.ينجز توصيل(جمع) منطقي أو ضم بتات bitwise على عاملين،ويرجع النتيجة.من أجل العمليات المنطقية(البولين Boolean) ستكون النتيجة صواب فقط إذا ما قيم معاملين لصواب.من أجل العمليات البتات(العددية الصحيحة)،كل بت(وحدة bit) خاص في النتيجة سيتم وضعها إلى 1 فقط إذا كان البتين(الوحدتين)المتطابقين في كلا معاملين هما واحد1.التركيب: *operand2* And *operand1*.مثال: isOfficer And isGentleman.
Or	الفصل Disjunction. ينجز فصل منطقي أو وحدوي(بتات) على عاملين،والعودة بنتيجة.من أجل العمليات المنطقية(البولينية)،ستكون النتيجة صح إذا ما تم تقييم أي من معاملين لصواب.من أجل عمليات البتات(العددية الصحيحة)،كل بت خاص في النتيجة سيتم وضعه لواحد1إذا كان البت الموافق في عامل 1.التركيب: operand1 Or operand2.مثال: enjoyMountains Or enjoySea.
AndAlso	توحيد قاصر الدارة Short-circuited conjunction. هذا المعامل مكافئ للنسخة المنطقية من المعامل And،ولكن إذا تم تقيم معامل الأول إلى خطأ،فإن معامل الثاني لن يتم تقيمه على الإطلاق.وهذا المعامل لايدعم عمليات البتات bitwise . التركيب:operand1 AndAlso operand2. مثال: isOfficer AndAlso isGentleman : .
OrElse	الفاصل قاصر الدارة Short-circuited disjunction. هذا المعامل مكافئ للنسخة المنطقية للمعامل Or،ولكن إذا تم تقيم معامل الأول إلى صح ،فإن معامل الثاني لن يتم تقيمه على الإطلاق.وهذا المعامل لايدعم عمليات البت bitwise (العددية الصحيحة) التركيب: operand1 OrElse operand2.مثال: enjoyMountains OrElse enjoySea.
Not	النفي Negation. ينجز نفي منطقي أو بتات على عامل مفرد.من أجل العمليات المنطقية(البولين)ستكون النتيجة صواب إذا تم تقييم معامل إلى خطأ،وخطأ إذا ما تم تقيم معامل إلى صح. ومن أجل عمليات البت(العددية الصحيحة)،كل بت خاص في النتيجة سيتم وضعه إلى 1 إذا ما كان بت معامل المطابق هو 0،ويتم وضعه إلى 0إذا ماكان بت معامل هو 1. التركيب: Not operand1.مثال: Not readyToSend.
Xor	استبعاد Exclusion. ينجز استثناء"أو (استثناء) exclusive or "على عاملين،ويعود بنتيجة.من أجل العمليات المنطقية(البولين)، ستكون النتيجة صح فقط إذا كان للعوامل قيم منطقية مختلفة(صواب أو خطأ True or False).من أجل عمليات البت(العددية الصحيحة)، كل بت خاص في النتيجة سيتم وضعه إلى 1 فقط إذا البتات المطابقة في معاملات مختلفة. التركيب: operand1 Xor operand2. مثال: chickenDish Xor beefDish.
<<	الرفع اليساري Shift left. المعامل رفع يساري Shift Left يرفع البتات للعامل الأول إلى اليسار بعدد المواضع المخصصة في معامل الثاني،ويعود بنتيجة. البتات المسحوبة للنهاية اليسرى للنتيجة يتم فقدانها lost،البتات المضافة إلى النهاية اليمنى هي دائماً 0.وهذا المعامل يعمل بشكل أفضل إذا كان معامل الأول قيمة عددية صحيحة بدون إشارة. التركيب: operand1 << operand2.مثال: 3 << &H25.
>>	الرفع اليميني Shift right. يرفع معامل الرفع اليميني Shift Right البتات للعامل الأول إلى اليمين بعدد المواضع المخصصة في العامل الثاني.ويعود بنتيجة. البتات المسحوبة للنهاية اليمنية للنتيجة يتم فقدانها(ضياعها)،والبتات المضافة إلى النهاية اليسرى دائما نفس البت الأصلي في الموضع اليساري على الأغلب.وهذا المعامل يعمل أفضل إذا كان عامل الأول قيمة عددية صحيحة بدون إشارة. التركيب: operand1 >> operand2.مثال: 2 >> &H25.

المعامل	الوصف
=	المساواةequals (المقارنة).(comparison).يقارن عاملين ويعود صواب إذا كانا متساويان في القيمة. التركيب: operand1 = operand2.مثال: expectedAmount = actualAmount.
<>	لايساويNot equals. يقارن عاملين ويعود بصواب True إذاكانا غير متساويين في القيمة. التركيب: operand1 <> operand2.مثال: startValue <> endValue.
<	أقل منLess than. يقارن عاملين ويعود بصواب إذا كان معامل الأول أقل في القيمة من الثاني. عند مقارنة قيم لسلسلة حرفية،فالراجع يكون صواب إذا ظهر معامل الأول أولاً عند ترتيب سلسلتين حرفيتين. التركيب: operand1 < operand2.مثال: raiseRate < inflationRate.
>	أكثر(أكبر Greater than)من.يقارن عاملين ويعود صح إذا كان الأول أكبر من الثاني في القيمة.وعند مقارنة قيم لسلسلة حرفية،العائد يكون صواب إذا ظهر معامل الأول أخيراً عند ترتيب سلسلتين حرفيتين. التركيب : operand1 > operand2. مثال: raiseRate > inflationRate.
<=	أقل من أو يساوي Less than or equal to.يقارن عاملين ويعود بصواب إذا كان الأول أقل أو يساوي بالقيمة معامل الثاني. التركيب: operand1 <= operand2.مثال: raiseRate <= inflationRate.
>=	أكبر من أو يساوي لـ Greater than or equal to.يقارن عاملين ويعود بصواب إذا كان الأول أكبر من الثاني أو يساويه في القيمة . التركيب: operand1 >= operand2. مثال: raiseRate >= inflationRate.
Like	نموذج(عينة) مقارنPattern comparison. يقارن عامل أول إلى نموذج مخصص في الطرف(عامل) الثاني،ويعود بصواب إذا كان هناك تطابق.يدعم طرف(عامل)العينة(النموذجPattern) بعض كروت التعريف wildcard (المحددة القيمة مسبقاً)القاعدية وخيارات مختارة،وتم شرحها كاملاً في المستندات المزودة بالفيجوال أستوديو.يتضمن الدوت نت ميزة تدعى التعابير النظامية *regular expressions*والتي توفر الكثير وأعظم النماذج الشاملة مطابقة(ملائمة) الحل solution. التركيب: operand1 Like operand2.مثال: governmentID Like ssnPattern.
Is	مقارن نوعType comparison. يقارن عامل الأول بكائن أخر object، نوع بيانات data typeأو "لاشيءNothing "،ويعود بصواب إذا كان هناك تطابق.وسأوثق هذا المعامل بالتفصيل أكثر فيما بعد في النص.وفي الفصل ا لثامن. التركيب: operand1 Is operand2.مثال: someVariable Is Nothing.
IsNot	مقارن نفي النوعNegated type comparison. وهذا المعامل يقطع(يقصر)دارة استخدام المعاملين Is وNotمع بعضهما.التعبيران التاليان متكافئان: first IsNot second ، Not (first Is second) التركيب: operand1 IsNot operand2.مثال: something IsNot somethingElse.
TypeOf	نسخة(حالة)مقارنةInstance comparison. يعود بنوع البيانات لقيمة أو متغير.نوع كل فئة أو نوع بيانات في الدوت نت يتم تنفيذها(تنفيذه)ككائن،بالاعتماد على System.Type. المعامل TypeOfيمكن أن يتم استخدامه مع المعامل Is. التركيب: TypeOf operand1 Is typeOperand.مثال: TypeOf someVariable Is Integer.
AddressOf	استعادة التفويضDelegate retrieval. يعود بتفويض(مشروح في الفصل الثامن)والتي تمثل حالة خاصة من طريقة إجراء. التركيب:AddressOf method1 . مثال: AddressOf one.SomeMethod.
GetType	النوع المسترجعType retrieval. يعود بنوع البيانات لقيمة أو متغير.تماماً مثل المعامل TypeOf.مهما يكن،يعمل GetTypeمثل دالة(وظيفة)،ولا يحتاج إلى أن يتم استخدامه مع المعامل Is. التركيب: GetType(operand1).مثال: GetType(one).

تستخدم المعاملات غير الإسنادية Non-assignment operators عاملاتها operandsللحصول على نتيجة،ولكن لا تسبب تبديل عوامل نفسها بأي طريقة.معامل الإسناد يعمل على تحديث العامل الذي يظهر في جهته اليسارية.بالإضافة إلى عامل الإسناد القياسي ،تتضمن الفيجوال بيسك عدة معاملات تجمع(تضم) معامل الإسناد مع بعض المعاملات الثنائية.يبين الجدول التالي قائمة بمعاملات الإسناد هذه.

المعامل Operator	معتمد على Based on
=	معامل الإسناد القياسي.
+=	+(إضافةaddition)
-=	-(طرحsubtraction)
*=	*(ضرب multiplication)
/=	/(قسمة division)
\=	\(قسمة عددي صحيحinteger division)
^=	^ (القوة(الأس) exponentiation)
&=	&(سلسلة(تتابع حرفي) concatenation).
<<=	<<(الرفع اليساري shift left)
>>=	>> (الرفع اليميني shift right)

معاملات الإسناد هذه هي مجرد اختصارات للمعاملات المجسدة الكاملة.على سبيل المثال،لإضافة 1إلى متغير رقمي،بإمكانك استخدام واحد من هاتين الطريقتين:

```
' ----- زيادة totalSoFar بـ 1 .
       totalSoFar = totalSoFar + 1
' ----- طريقة  أخرى لـزيادة totalSoFar بـ 1 .
       totalSoFar += 1
```

المتغيرات الستاتيكية Static Variables

عادة ينتهي عمر المتغير على مستوى الإجراء المحلي عندما ينتهي الإجراء.ولكن في بعض الأحيان ومن المحتمل أنك تريد من متغير أن يحفظ قيمته بين كل استدعاء ضمن الإجراء.وبعض الأحيان أيضاً ممكن أن تحتاج إلى مليون دولار ولكن لاتستطيع الحصول عليها دائماً.ولكن بإمكانك الحصول على متغيرات تحفظ قيمها إذا أردت.وهي تدعى المتغيرات الستاتيكية.للتصريح عن متغير ستاتيكي*static variables* ، استخدم الكلمة المحجوزة Staticفي مكان الكلمة المحجوزة Dim.

```
Static keepingTrack As Integer = 0
```

إن إسناد المتغير إلى صفر يتم عمله مرة واحدة فقط.وعندما تعمل على إنشاء حالة(نسخة) من نوع يحتوي على هذه العبارة.فيما بعد.فإنه يحفظ القيمة التي تم إسناده له من قبل مهما تكن حتى يتم تدمير الحالة(النسخة).يمكن أن يتم إنشاء المتغيرات الستاتيكية فقط ضمن الإجراءات.

المصفوفات Arrays

تعمل التطبيقات البرمجية غالباً مع مجموعات من البيانات المترابطة،وليس فقط مع قيم البيانات المعزولة.تتضمن الفيجوال بيسك طريقتين رئيسيتين للعمل مع مثل هذه المجموعات من البيانات:التجمعاتcollections (سنناقشها فيما بعد) والمصفوفاتarrays.تعمل المصفوفة على إسناد مواقع عددية إلى كل بند مضمن(محتوى) في المجموعة،تبدأ بـ0 وتنتهي بـأقل1 من عدد البنود المضمنة(المحتواة).فمصفوفة من خمس بنود لديها عناصر مرقمة من 0 إلى 4.وكمثال ،تصور أنك تبرمج تطبيق يحاكي حديقة حيوانات.يمكن أن تضمن مصفوفة مسماة الحيواناتanimalsوالتي تتضمن اسم كل حيوان في حديقتك:

. حيوان #0:آكل النمل Aardvark
. حيوان #1: القرودBaboon
. حيوان #2:الشمبانزي Chimpanzee
. حيوان #3:الحمار Donkey
. وهلم وجر .

تعرف الفيجوال بيسك عناصر المصفوفة array elements بالعدد الموضوع ضمن أقواس parenthesized number بعد اسم المصفوفة.فمن أجل حيواناتنا،يعمل إسناد بسيط على وضع اسم نصي لكل حيوان في عنصر المصفوفة.

```
animal(0) = "Aardvark"
animal(1) = "Baboon"
animal(2) = "Chimpanzee"
animal(3) = "Donkey"
```

استخدام كل عنصر مصفوفة سهل جداً:

```
MsgBox("The first animal is: " & animal(0))
```

فكل عنصر من المصفوفة لايختلف كثيراً عن المتغير القائم بذاته.في الحقيقة،بإمكانك فقط اعتبار مجموعة من الحيوانات في كود المثال لأن تكون متغيرات واضحة:متغير مسمىanimal(0)، والآخر مسمى animal(1)،وهكذا.ولكن هي أفضل من المتغيرات العادية لأنك تستطيع معالجتهم جميعاً كمجموعة.على سبيل المثال،تستطيع تفحص كل عنصر باستخدام الحلقة For...Next.خذ مصفوفة عددية صحيحة مسماة "كل بندeachItem:" مع عناصر مرقمة من 0إلى 2. يضيف مقطع الكود التالي البنود المستقلة للمصفوفة وكأنهم متغيرات مختلفة .

```
Dim totalAmount As Integer
        totalAmount = eachItem(0) + eachItem(1) + eachItem(2)
```

ولكن بما أن البنود في مصفوفة عددية،تستطيع استخدام الحلقة For...Next لعمل مسح لكل العناصر،بالدور(واحد واحد)

```
Dim totalAmount As Integer = 0
        For counter As Integer = 0 To 2
            ' ----- Keep a running total of the items.
            totalAmount += eachItem(counter)
        Next counter
```

قبل أن تسند قيم إلى عناصر المصفوفة،أو استخلاص هذه العناصر،عليك التصريح وتحجيم المصفوفة التي تريد.تعمل عبارة Dimعلى إنشاء مصفوفة كما تعمل مع المتغيرات العادية،والعبارة ReDimتعمل إعادة تحجيم المصفوفة بعد أن تكون موجودة مسبقاً.

```
Dim animal(0 To 25) As String ' 26-element array
        Dim moreAnimals() As String ' An undefined String array
        ReDim moreAnimals(0 To 25) ' Now it has elements
```

عادة ،ستعمل العبارة ReDimعلى مسح wipeأي بيانات موجودة ومخزنة في كل عنصر لمصفوفة.إضافة الكلمة المحجوزة Preserveتعمل على صيانة retainsجميع البيانات.

```
ReDim Preserve moreAnimals(0 To 30) ' Keeps elements 0 to 25
```

كل عنصر في المصفوفة كائن مستقل والذي يمكن أن يتم إسناد بيانات له عند الحاجة.في هذا المثال،كل عنصر هو سلسلة حرفية String،ولكن تستطيع استخدام نوع ذو قيمة أو نوع مرجعي ترغب به في التصريح عن المصفوفة.إذا أنشئت مصفوفة من عناصر كائنية،بإمكانك دمج ومطابقة البيانات في المصفوفة،فالعنصر 0لايحتاج أن يحتوي نوع بيانات كما هو الحال في العنصر1.
المصفوفة نفسها هي أيضاً كائن مستقل-حالة(نسخة) فئة والتي تدير مجموعتها من العناصر المحتواة.إذا أردت تخصيص كامل المصفوفة،وليس فقط واحد من عناصرها(وكان هناك وقت عند الحاجة لعمل هذا)،استخدم اسمها دون أي قوس أو قيم موضوعة بترتيب معين(مرتبة).

المصفوفات المتعددة الأبعاد Multidimensional Arrays

تدعم مصفوفات الفيجوال بيسك أكثر من بعد واحد*dimension* one (أو "رتبة rank ").تشير الأبعاد dimensionsإلى عدد المجالات المستقلة المدعومة بالمصفوفة. فمصفوفة وحيدة البعدone-dimensional array، مثل مصفوفة الحيوان animalالمذكورة سابقاً،تتضمن مجال وحيد.أما المصفوفة ذات بعدين تتضمن مجالين مفصولين بفاصلةcomma-delimited ranges، تشكل شبكة منظمة من العناصر،مع مجالات مفصولة من أجل الصفوف والأعمدة.

```
Dim ticTacToeBoard(0 To 2, 0 To 2) As Char ' 3 x 3 board
```

يمكن أن يكون لمصفوفة حتى 60 بعد مختلف،على الرغم من أنه توجد طرق أفضل لتنظيم البيانات من تقسيمها ضمن عدة أبعاد.

حدود المصفوفة Array Boundaries

الحد الأقل لبعد أي مصفوفة هو0،وكما أشرت بالشرط 0 To x عند تعريف أو إعادة تحجيم المصفوفةredimensioning. بإمكانك عملياً إزالة الجزء من العبارة"0 To"،وتعمل على تضمين فقط الحد الأعلىupper bound .

```
' ----- These two lines are equivalent.
        Dim animal(0 To 25) As String
        Dim animal(25) As String
```

هاتان العبارتان كلاهما يعملان على إنشاء مصفوفة بـ26 عنصر.مرقمة من 0 إلى25.يوجد العديد من الحالات الخاصة حيث يتم السماح للحدود الأدنى غير صفر،كالعمل مع المصفوفات الأقدم المولدة عن تقنية برمجة المكونات COM-generated arrays.ولكن تركيب التصريح القياسي

في الفيجوال بيسك لايسمح لك بإنشاء مصفوفات مع حدود دنيا غير الصفر(أي دائماً الحد الأدنى هو الصفرnonzero lower bounds).لتحديد الحد الأدنى والأعلى الحالي لبعد مصفوفة،استخدم الدالة LBoundوالدالةUBound.

```
MsgBox("The board is " & (UBound(ticTacToeBoard, 1) + 1) & _
   " by " & (UBound(ticTacToeBoard, 2) + 1))
```

إذا كانت مصفوفتك تحتوي فقط على بعد واحد،فليس عليك أن تخبر عن الحد الأدنى أو الحد الأعلى لأي بعد تريد أن تتحقق منه.

```
MsgBox("The upper element is numbered " & UBound(animal))
```

تتضمن كل مصفوفة أيضاً الطريقة GetLowerBoundوالطريقة GetUpperBoundواللتان تعودان بنفس النتيجة التي يعود بها كل من LBoundو UBound على الترتيب.مهما يكن رقم البعد الذي تمرره إلى الطريقة GetLowerBoundوالطريقة GetUpperBoundيبدأ من الصفر0،بينما القيم بالنسبة LBound وUBoundتبدأ العد من 1.

```
MsgBox("The board is " &(ticTacToeBoard.GetUpperBound(0) + 1) & _
" by " & (ticTacToeBoard.GetUpperBound(1) + 1))
```

إسناد قيم أولية(تمهيد)للمصفوفات

حالما تصرح عن عناصر مصفوفتك،بإمكانك تخزين واستخراج العناصر عندما تحتاج لذلك.ومن الممكن أيضاً تخزين العناصر في مصفوفتك حالما تصرح عنها.وتظهر قائمة العناصر الجديدة في مجموعة ضمن حاصرتين(قوسين مدببينcurly braces)

```
Dim squares( ) As Integer = {0, 1, 4, 9, 16, 25}
```

عليك عدم وضع مواصفتي الحد الأدنى والأعلى lower and upper bound specifications عند إنشاء مصفوفة بهذه الطريقة.فمصفوفة "المربعات squares" المبينة هنا سيكون لها عناصر مرقمة من 0 إلى 5.

الأنواع "بدون قيمة" Nullable Types

متغيرات الأنواع ذات القيمة تعمل بشكل جيد جداً،تحفظ قيم بياناتها مدة عمرها.والأنواع المرجعية تعمل أيضاً بشكل جيد،ولكن يمكن ملئها بلاشيءNothing. وهذا الاختلاف قد أصبح شوكة في جانب الأنواع ذات القيمة.ولكن مع الإصدار فيجوال بيسك2008،أصبحت الأنواع ذات القيمة الآن من الممكن إسناد لها لاشيءNothing . وهذا النوع الجديد "بدون قيمة *nullable* " أساسي عندما تريد أن تكون لديك حالة غير محددة من أجل نوع ذو قيمة قياسي(وهو مفيد خاصة عند العمل مع حقول قواعد البيانات).خذ هذه الفئة والتي تدير معلومات موظف:

```
Public Class Employee
        Public Name As String
        Public HireDate As Date
        Public FireDate As Date
        Public Sub New(ByVal employeeName As String,ByVal dateHired As Date)
            Me.Name = employeeName
            Me.HireDate = dateHired
        End Sub
    End Class
```

تعمل هذه الفئة بشكل جيد،ما عدا "تاريخ الطرد FireDate " فهو ليس صحيح.بشكل افتراضي،سيتم وضع تاريخ الطرد FireDateإلى1كانون الثاني ،سنة1بعد الميلاد،12:00، وتستطيع استخدام ذلك التاريخ وكأنه تاريخ "عدم الطرد".ولكن ماذا سيحدث إذا ما طردت شركتك أحدما تماماً في تلك اللحظة ،هل سيكون قد مرّ قرنين؟لحل مثل هذه القضية،تتيح لك الأنواع "بدون قيمة nullable " من إسناد واستخراج لاشيء من متغيرات للنوع ذو القيمة. هذا الفيتامين يغني enrichedالأنواع ذات القيمة ويتم التصريح عنه باستخدام تركيب علامة الاستفهام الخاصة.

```
'أي من هاتين العبارتين سيعمل بشكل جيد
    Public FireDate As Date?
    Public FireDate2? As Date
```

كل من هاتين العبارتين سيعمل ولكني أفضل العبارة الأولى،بحيث يتم إضافة علامة الاستفهام إلى لنوع البيانات.وحالما يتم التصريح.يمكن للنوع ذو القيمة أن يأخذ إما بيانات قياسية أو لاشيء.

```
  FireDate = Nothing
  FireDate = #7/18/2008#
If (FireDate Is Nothing) Then...
```

يوجد تركيب خاص عند تعريف أنواع ذات قيمة خاصة بك كـ"بدون قيمةnullable "،ولكن بما أنها تستخدم مواصفة الفيجوال بيسك" generics "،لذا سأنتظر حتى الفصل 16 لأقدمها لك.

دوال الفيجوال بيسك المشتركة Common Visual Basic Functions

في هذا المقطع الأخير سأعمل على تضمين قوائم بالدوال الجاهزة في الفيجوال بيسك،والعديد منها ستستخدمه بانتظام في تطبيقاتك.وأيضاً ما تم جدولته هنا بعض من أعضاء مكتبة فئة إطار العمل Framework Class Library (FCL)والتي تكرر مواصفات كانت جزء من لغة الفيجوال بيسك قبل الدوت نت،ولكن تم نقلها إلى إطار العمل من أجل إمكانية وصول عامة.

دوال التحويل Conversion Functions

تتيح لك دوال التحويل من تحويل واحد من نوع بيانات الفيجوال بيسك إلى نوع آخر.وليس لك الحرية الكاملة من أجل الكل،لذلك لا تذهب بعيداً وتعمل على تحويل النص"مرحباً" إلى قيمة عددية صحيحة وتتوقع منها أن تعمل.ولكن تحويل الأعداد من نوع عددي إلى أخر ،أو تحويل الأعداد بين الأنواع العددية والسلاسل الحرفية يعمل بشكل عام جيداً.كل هذه العبارات (ماعدا CType)لديها نفس التركيب:

dest = CXxxx(source)

حيث source هو القيمة التي يجب تحويلها بواسطةCXxxx. ليس عليك إسناد النتيجة إلى متغير ما *dest*، تستطيع استخدام النتيجة في أي مكان ترغب به كما هو الحال مع قيم المتغيرات والقيم الحرفية،يبين الجدول التالي دوال التحويل.

الدالة Function	الوصف Description
CBool	يحول قيمة ما إلى منطقي Boolean
CByte	يحول قيمة ما إلى بايت Byte
CChar	يحول قيمة ما إلى حرف Char. إذا كانت القيمة المصدرية هي من نوع سلسلة حرفيةstring ،سيتم تحويل فقط الحرف الأول.
CDate	يحول قيمة ما إلى تاريخ Date. إذا كانت القيمة المصدرية هي سلسلة حرفية.فيجب أن تكون في تنسيق وقت أو تاريخ صحيح.
CDbl	يحول قيمة ما إلى مزدوج Double.

CDec	يحول قيمة ما إلى عشري Decimal.
CInt	يحول قيمة ما إلى عددي صحيح Integer.
CLng	يحول قيمة ما إلى طويل Long.
CObj	يحول قيمة ما إلى كائنObject. و هذا مفيد عند تريد تخزين نوع قيمة ما كحالة كائنObject .
CSByte	يحول قيمة ما إلى بايت بإشارة SByte.
CShort	يحول قيمة ما إلى قصير Short.
CSng	يحول قيمة ما إلى مفرد Single.
CType	يحول قيمة ما إلى أي نوع محدد،فئة، أو واجهة،إما في تطبيقك أو في مكتبة فئة إطار العملFCL . والتركيب هو: `CType(sourceData, newType)` حيث هو *newType* نوع بيانات،على سبيل المثال:

```
        Dim x As String
    x = CType(5, String)
```

تحويل العدد الصحيح 5إلى سلسلة حرفية.وكما مع دالات التحويل الأخرى،لاتستطيع تحويل البيانات من نوع إلى آخر إذا كانت الأنواع متضاربة،أو لو لم يوجد تحويل متاح بحيث يعرف كيف يعمل على إنتاج النوع المستهدف من النوع المصدري.

CUInt	يحول قيمة ما إلى عددي صحيح بدون إشارةUInteger .
CULng	يحول قيمة ما إلى طويل بدون إشارةULong .
CUShort	يحول قيمة ما إلى قصير بدون إشارة UShort.

الدوال الخاصة بالتاريخ Date-Related Functions

تتضمن الفيجوال بيسك العديد من الدوال المصممة لإدارة قيم التاريخ والوقت.يجدول الجدول التالي هذه الدوال.معظم هذه الدوال تقبل معامل نسبيي arguments مصدري واحد أو أكثر،وتعمل على إرجاع إما تاريخDate، سلسلة حرفية String أو نتيجة عددية numeric result.

الدالة Function	الوصف Description
DateAdd	يضيف أو يطرح قيمة تاريخ أو وقت إلى تاريخ البدء.مثلاً،تستطيع إضافة 12دقيقة،أو طرح ثلاث سنوات من تاريخ معطى
DateDiff	تعود بالفرق بين قيمتي وقت أو تاريخ.تستطيع تخصيص الفترة،مثل شهر،أو ثانية.
DatePart	تعود بمكون واحد لتاريخ أو وقت.مثل ساعة أو تعود بسنة .
DateSerial	تعود بتاريخ Dateمبني من قيم شهر أو يوم أو سنة.
DateString	تعود بالتاريخ الحالي كسلسلة حرفية.تستطيع أيضاً وضع التاريخ على كمبيوتر محلي باستخدام هذه الكلمة المحجوزة.
DateValue	تعود بحصة تاريخية من قيمة وقت وتاريخ موحد(مدمج)،ويتم طرح(إخراج)الحصة(قسم)الوقت(لايتم تضمينه)
Day	تعود باليوم من قيمة تاريخ معطى.
FormatDateTime	تعمل على تنسيق تاريخ معطى أو وقت كسلسلة حرفية.باستخدام تنسيقات مجموعة صغيرة أو محددة مسبقاً.لقد تم تضمين هذه الكلمة المحجوزة بالتوافق مع كود صيغة الفيجوال بيسك المصغرة الأقدم VBScript code.
Hour	تعود بالساعة من قيمة وقت معطى.
IsDate	تشير فيما إذا التاريخ المزودة به هذه الدالة هو تاريخ صحيحvalid date .
Minute	تعود بالدقيقة من قيمة وقت معطى.
Month	تعود بالشهر من قيمة تاريخ معطى.
MonthName	تعود باسم الشهر من قيمة شهرية عددية،من 1 إلى 12.
Now	تعود بالتاريخ والوقت الحالي.مكافئة لـ TimeOfDay.
Second	تعود بالثواني من قيمة وقت معطى.
TimeOfDay	تعود بالتاريخ والوقت الحالي.مكافئة Now.
Timer	تعود بعدد الثواني التي مرت منذ منتصف ليل اليوم الحالي.ويتم إعادة وضع هذه الدالة إلى 0كل منتصف ليلة.
TimeSerial	تعود بتاريخ Dateمبني من قيم ساعات خاصة أو دقائق أو ثواني.
TimeString	تعود بالوقت الحالي كسلسلة حرفية.تستطيع أيضاً وضع الوقت الذي على كمبيوتر محلي باستخدام هذه الكلمة المحجوزة.
TimeValue	تعود بحصة الوقتportion من قيمة تاريخ ووقت مدمج،وحصة التاريخ يتم طردها(إزالتها)discarded).
Today	تعود بالتاريخ الحالي.
Weekday	تعود بعدد صحيح يشير إلى اليوم من الأسبوع.
WeekdayName	تعود باسم اليوم من الأسبوع بالنسبة ليوم عددي صحيح من أسبوع.
Year	تعود بالسنة من قيمة تاريخ معطى.

المتغيرات التي تم إنشاءها كـ System.DateTime (أو تاريخ من فيجوال بيسك)كلها تتضمن العديد من الخاصيات والطرق والتي توفر مواصفات مشابهة للدوال المجدولة في الجدول السابق،على سبيل المثال،الخاصية "ثانية" تعود بعدد الثواني.

```
Dim meetingTime As Date
        meetingTime = #11/7/2005 8:00:03 AM#
        MsgBox(meetingTime.Second) ' Displays '3'
        MsgBox(Second(meetingTime)) ' Also displays '3'
```

تستطيع إما استخدام دوال الفيجوال بيسك الجوهرية(القاعدية intrinsic) أو ما يكافئها من طرق وخاصيات System.DateTime في كودك .وكلا التقنيتان توفران نفس النتيجة.

الدوال العددية Numeric Functions

تتضمن الفيجوال بيسك الكثير من الميزات للعمل على نحو مدهش مع الأعداد،بالإضافة إلى المعاملات التي تعالج البيانات،يجدول الجدول التالي العديد من الدوال الخاصة بالأعداد.

الدالة Function	الوصف Description
Fix	تبتر القسم(الحصة)العشرية من عدد ما،وتعود فقط بالحصة الصحيحة.مشابهة للدالة Int.

FormatCurrency	تنسق عدد معطى كقيمة عملةcurrency .باستخدام تنسيقات لمجموعة صغيرة معرفة مسبقاً.وهذه الخاصية تم تضمينها من أجل التوافق العائد لكود صيغة فيجوال بيسك المصغرة VBScript الأقدم.
FormatNumber	تنسق عدد معطى كعدد عام.باستخدام تنسيقات لمجموعة صغيرة معرفة مسبقاً.وهذه الخاصية تم تضمينها من أجل التوافق العائد لكود صيغة فيجوال بيسك المصغرة الأقدم.
FormatPercent	تنسق عدد معطى كنسبة مئوية.باستخدام تنسيقات لمجموعة صغيرة معرفة مسبقاً.وهذه الخاصية تم تضمينها من أجل التوافق العائد لكود صيغة فيجوال بيسك المصغرة الأقدم.
Int	تعود بعدد صحيح والذي يكون أقل من أو يساوي القيمة المزودة له.مشابهة للدالة Fix.
IsNumeric	تشير فيما إذا البيانات الموفرة لهذه الدالة هي عدد محقق valid number.
Oct	تنسق عدد ما كثمانيoctal .وتعود بالتمثيل النصي لهstring representation .
Val	تستخرج العدد المحقق الأول first valid number من نص stringوترجع به.

يتضمن إطار عمل الدوت نت الفئة System.Math.والتي تحتوي العديد من أعضاء الدوال المتعلقة بالرياضيات.بعض منها،مثل Round، Sin و Log، تم تنفيذها كدوال جوهرية في الفيجوال بيسك6،ولكن تم نقلها من اللغة إلى الفئة Mathفي الدوت نت.
تتضمن الفيجوال بيسك أيضاً العديد من الدوال المستخدمة الحسابات المالية financialوالإحصائية(أو المحاسبةaccounting).وكانت هذه الدوال مضمنة في الفيجوال بيسك6.وبما أنها ليست على علاقة بالمشروع الذي ناقشه في هذا الكتاب ،لذلك سأعمل فقط على جدولت أسمائها: DDB, FV, ،.IPmt, IRR, MIRR, NPer, NPV, Pmt, PPmt, PV, Rate, SLN, and SYD

دوال السلسلة الحرفية String Functions

معالجة السلاسل الحرفية هو جزء جوهري من برمجة ويندوز.ميزة XMLالجديدة المضمنة مع الدوت نت هي إجراءات روتينية -string manipulation routines لمعالجة النصوص بشكل رائع ،على الرغم من التعقيدات المخفية hidden from view التي معها.
تتضمن الفيجوال بيسك العديد من الدوال المصممة لمعالجة السلاسل الحرفية stringsوالحروفcharacters .وهي مجدولة في الجدول التالي وكما مع معظم الدوال الأخرى،هذه الدوال تعود بنص جديد أو قيمة،تترك النص الأصلي أو القيم المصدرية سليمةintact .والاستثناء الوحيد(الأعزلlone) هو عبارة Mid، والتي تعدل القيمة المصدرية لمتغيرsource variable's value .

الدالة Function	الوصف Description
Asc, AscW	تعود بالقيمة العددية آسكي ASCIIأو يونيكود Unicodeلحرف ما.
Chr, ChrW	لعدد معطى،فإن هذه الدوال تعود بما يطابق هذا العدد من حروف الآسكي ASCIIأو اليونيكود Unicode.
Filter	تعود بمصفوفة والتي هي مجموعة جزئية من مصفوفة مصدرية،ولكن تضمن فقط العناصر التي تطابق نموذج ما.
Format	تنسق قيم عددية،أو تاريخ أو وقت باستخدام أكواد تنسيق خاصة أو معرفة مسبقاً.
GetChar	تستخرج حرف وحيد من سلسلة حرفية أكبر.
InStr	تعود بموضع مجموعة جزئية ضمن سلسلة حرفية أكبر.
InStrRev	تعود بموضع مجموعة جزئية ضمن سلسلة حرفية أكبر،البحث من نهاية السلسلة الحرفية حتى البداية.
Join	تعود بسلسلة حرفية stringمبنية من سلسل concatenation مصفوفة من السلاسل الحرفيةarray of strings .
LCase	تحول سلسلة حرفية لما يكافئها ولكن بحالة الأحرف الصغيرة lowercase equivalent.
Left	تعود بالحصة اليسارية leftmost من سلسلة حرفية.
Len	تعود بطول سلسلة حرفية.
LSet	تعمل تجانب لليسار ضمن سلسلة حرفية أكبر بالمساحة(الفراغ).
LTrim	تزيل المسافات من بداية السلسلة الحرفية.
Mid	تستخرج مجموعة نصية جزئية من وسط middleسلسلة حرفية أكبر.
Mid	عبارة تعدل مجال من الأحرف في سلسلة حرفية موجودة بمحتوى جديد.وهذه ليست دالة،ولكن عبارة فيجوال بيسك خاصة.وتركيبها يختلف كثيراً عن معظم ميزات الفيجوال بيسك.
Replace	تبدل حالات من مجموعة نصية جزئية بمجموعة جزئية أخرى،والكل ضمن سلسلة حرفية أكبر.
Right	تعود بالحصة اليمنية rightmostلسلسلة حرفية .
RSet	تعمل تجانب لليمين ضمن سلسلة حرفية أكبر بالمسافات(الفراغات spaces).
RTrim	تزيل المسافات من نهاية سلسلة حرفية.
Space	تولد سلسلة حرفية تحوي عدد محدد من محارف الفراغspace characters .مشابهة للدالة StrDup.
Split	تقسم نص إلى مصفوفة من مجموعات نصية فرعية بالاعتماد على محدد معين delimiter.
Str	تحول عدد ما إلى تمثيله النصي .
StrComp	تقارن سلسلتين نصيتين ،وتعود بعدد صحيح يشير إلى أمر ترتيبهما sort order.
StrConv	تحول نص ما إلى تنسيق جديد بالاعتماد على كود التحويل.بعض التحويلات تتضمن تغير حالة المحتوى.
StrDup	تولد نص يحتوي عدد مخصص من حرف معين.مشابهة للدالة Space.ولكن تعمل مع أي حرف.
StrReverse	تعكس(تقلب) الحروف في سلسلة حرفية.
Trim	تزيل الفراغات من بداية ونهاية سلسلة حرفية.
UCase	تحول سلسلة حرفية لما يكافئها ولكن بحالة الأحرف الكبيرة uppercase equivalent.

المتغيرات المنشئة كـ System.String (أو سلاسل حرفية للفيجوال بيسك).كل منها يتضمن العديد من الخاصيات والطرق والتي توفر ميزات مشابهة للدوال المجدولة في الجدول السابق،على سبيل المثال،تعود الخاصية Lengthبعدد الأحرف في سلسلة حرفية.

```
Dim simpleString As String = "abcde"
        MsgBox(simpleString.Length)  ' Displays '5'
        MsgBox(Len(simpleString))  ' Also displays '5
```

بإمكانك استخدام إما دوال الفيجوال بيسك القاعدية intrinsic Visual Basic functions أو ما يكافئها من طرق وخاصيات System.String في كودك.كل تقنية توفر نفس النتيجة.على الرغم من أن تفاصيل التركيب والخيارات يمكن أن تتنوع(تختلفvary).

دوال أخرى Other Functions

تتضمن الفيجوال بيسك العديد من الدوال التي ترفض أن تكون مقحمة squeezedفي أي من التصنيفات السابقة.يجدول الجدول التالي قائمة بهذه الدوال.

الدالة Function	الوصف Description
DirectCast	تحول قيمة ما من نوع بيانات إلى أخر.و يجب تحديد نهاية وبداية نوع البيانات.مشابهة للدوال TryCastو CType.
ErrorToString	تعود بتمثيل نصي لكود خطأ.وتعمل فقط مع أكواد خطأ النظام التي كانت متاحة سابقاً في الفيجوال بيسك6،وهذه الأكواد ماتزال متاحة في الدوت نت.
IsArray	تشير فيما إذا البيانات المزودة لها هي مصفوفة محققة(صحيحةvalid array).
IsDBNull	تشير فيما إذا البيانات المزودة لها هي قيمة قاعدة بيانات "بدون قيمة NULL ".
IsError	تشير فيما إذا البيانات المزودة بها حالة خطأ error condition.
IsNothing	تشير فيما إذا البيانات المزودة بها غيرمحددة،أو تم وضعها لـلاشيء Nothing.
IsReference	تشير فيما إذا البيانات المزودة بها هي نوع مرجعي أو نوع ذو قيمة.
QBColor	تعود بكود اللون من مجموعة صغير من الألوان المعرفة مسبقاً.
RGB	تعود بكود اللون المبني من المكونات المستقلة للأحمر،الأخضر،والأزرق.
SystemTypeName	اسم نوع بيانات الفيجوال بيسك المعطى،تعود هذه الخاصية بالاسم المكافئ لنوع بيانات الدوت نت.
TryCast	تحول قيمة من نوع بيانات لأخر.ويجب تحديد نوع بيانات البداية والنهاية.مشابهة للدوال DirectCast وCType.
TypeName	تعود باسم نوع البيانات الذي يلخص نوع بيانات المحتوى الموفر.والنص المعاد هو ملخص مجملgeneralized summary،وليس بالضرورة اسم نوع البيانات "صواب true "
VarType	تعود بكود يشير إلى نوع البيانات العام للمحتوى المزود به.

مشروع Project

سنستخدم في هذا الفصل ميزات كل من الدوال ونوع البيانات التي درسناها ،كل الكود في هذا الفصل سيظهر في وحدة برمجية للفيجوال بيسك مسماة "عامةGeneral " وسيتم تخزينها باسم General.vb.
لقد عملت على إضافة ملف مع وحدته البرمجية وتعريف الوحدة البرمجية يظهر كما يلي(افتح قالب مشروع ويندوز واعمل على إضافة وحدة برمجية له من قائمة مشروعPROJECT>>إضافة وحدة برمجية module وسمها *General.vb*)وأضف في بداية تعريف الوحدة البرمجية الكلمة المحجوزة Friend.

```
Friend Module general

End Module
```

كما ذكرت كل الكود الذي سنضيفه في هذا الفصل سيظهر ضمن هذين السطرين.تذكر أن الوحدات البرمجية تشبه كثيراً الفئات والتراكيب،ولكن لاتستطيع إنشاء حالة منها،فكل أعضاءها تتشارك مع جميع أجزاء كودك المصدري.وهذا ما يتيح لها الاستخدام في أي مكان في التطبيق.لانحتاج لعمل أي شيء خاص لجعلها متاحة لكامل البرنامج. إلا وضع إمكانية الوصول لكل عضو عند الحاجة.في البداية سنعمل على إضافة بعض الثوابت العامة المستخدمة على طول البرنامج.أضف الكود التالي بين السطرين السابقين كما يلي:

```
Friend Module general
    ' ----- Public constants.
    Public Const ProgramTitle As String = "The Library Project"
    Public Const NotAuthorizedMessage As String = _
        "You are not authorized to perform this task."
    Public Const UseDBVersion As Integer = 1
    ' ----- Constants for the MatchingImages image list.
    Public Const MatchPresent As Integer = 0
    Public Const MatchNone As Integer = 1
    Public Enum LookupMethods As Integer
        ByTitle = 1
        ByAuthor = 2
        BySubject = 3
        ByKeywordAny = 4
        ByKeywordAll = 5
        ByPublisher = 6
        BySeries = 7
        BySeriesID = 8
        ByBarcode = 9
        ByDatabaseID = 10
        ByAuthorID = 11
        BySubjectID = 12
        ByPublisherID = 13
    End Enum
    ' ----- Security values.
    Public Enum LibrarySecurity As Integer
        ManageAuthors = 1
        ManageAuthorTypes = 2
        ManageCopyStatus = 3
        ManageMediaTypes = 4
        ManageSeries = 5
        ManageGroups = 6
        ManageItems = 7
        ManagePatrons = 8
        ManagePublishers = 9
        ManageValues = 10
        ManageUsers = 11
        ProcessFees = 12
```

```
        ManageLocations = 13
        CheckOutItems = 14
        CheckInItems = 15
        AdminFeatures = 16
        DailyProcessing = 17
        RunReports = 18
        PatronOverride = 19
        ManageBarcodeTemplates = 20
        ManageHolidays = 21
        ManagePatronGroups = 22
        ViewAdminPatronMessages = 23
    End Enum
    Public Const MaxLibrarySecurity As LibrarySecurity = LibrarySecurity.ViewAdminPatronMessages

End Module
```

هذه الثوابت و العدادات توضح نفسها ،سيتم استخدام UseDBVersionلضمان أن التطبيق يطابق قاعدة البيانات المستخدمة عند إتاحة عدة إصدارات منها.

الثوابت MatchPresent و MatchNoneسيتم استخدامها لعمليات بحث بنود المكتبة.يعرف العدادان الأكواد التي تعين نوع بحث بنود المكتبة لإنجاز (طرق بحث LookupMethods)،و أكواد الأمن تستخدم لحصر الميزات التي سيكون مدير ما administratorقادر على عملها في التطبيق (LibrarySecurity)

حان الوقت لإضافة بعض الطرق.الطريقة الأولىCenterText ،،تمركز سطر من النص ضمن عرض معين.للمثال،إذا كان لديك نص " Hello, World"(12حرف في الطول) وأردت وضعه في مركز سطر يمكن أن يكون بطول 40حرف،ستحتاج إلى إضافة 14فراغ لبداية السطر(محددة بطرح 12من40،ومن ثم تقسيم النتيجة على 2).يستخدم الإجراء زوج من دوال الفيجوال البيسك الخاصة بالسلاسل الحرفية(مثل Trim، Left و Len)لمعالجة واختبار البيانات،ومعامل القسمة الصحيحة \ للمساعدة على حساب عدد الفراغات التي يجب إدخالها.

```
Public Function CenterText(ByVal origText As String, ByVal textWidth As Integer) As String
'جعل قطعة من النص في مركز السطر ذو اتساع معين,وإذا كان النص طويل يتم بتره
        Dim resultText As String
        resultText = Trim(origText)
        If (Len(resultText) >= textWidth) Then
            ' البتر عند الحاجة
            Return Trim(Left(origText, textWidth))
        Else
            ' إضافة فراغات في البداية و النهاية
            Return Space((textWidth - Len(origText)) \ 2) & resultText
        End If
    End Function
    Public Function LeftAndRightText(ByVal origLeft As String, ByVal origRight As String, _
            ByVal textWidth As Integer) As String
        ' بناء نص بالاتساع المحدد ولديه نص مضبوط على يمين ويسار السطر
        Dim leftText As String
        Dim rightText As String
        leftText = Trim(origLeft)
        rightText = Trim(origRight)
        If (Len(leftText & rightText) >= textWidth) Then
            Return Trim(Left(leftText & rightText, textWidth))
        Else
            Return leftText & Space(textWidth - Len(leftText & rightText)) & rightText
        End If
    End Function
```

تبدأ الدالةCenterText بعمل نسخة من النص الأصلي(origText)،تزيل أي فراغات إضافية بالدالة Trim.ومن ثم تختبر تلك النتيجة لرؤية فيما إذا ستتناسب على السطر.فإذا لم تكن كذلك،فإنها تقطع chops off حروف التذيل والتي تكون زائدة(غير متناسبة)،وتعود بتلك النتيجة.من أجل النصوص التي تتناسب والسطر textWidthاتساع الحروف،تضيف الدالة عدد مناسب من الفراغات لبداية النص،وتعود بالنتيجة.يضيف الكود السابق دالة مسماة LeftAndRightText.تعمل تماماً مثل الدالة CenterText،ولكنها تضع سلسلتين نصيتين مختلفتين عند أقصى النهايات اليسرى واليمنى لسطر النص.

لننتقل الآن إلى الجزء الثالث من الكود،يضيف هذا الجزء إجراء مسمى DigitsOnly.يعمل على بناء نص جديد مركب من الأرقام فقط الموجودة في النص المصدري، origText،وهو يعمل هذا باستدعاء الدالة IsNumeric لكل حرف في origText،واحد واحد.فكل رقم موجود يتم سلسلته(إلحاقة بالتسلسل)إلى نهاية destText.

```
Public Function DigitsOnly(ByVal origText As String) As String
        ' إعادة أرقام موجودة في نص فقط
        Dim destText As String
        Dim counter As Integer
        ' اختبار كل حرف
        destText = ""
        For counter = 1 To Len(origText)
            If (IsNumeric(Mid(origText, counter, 1))) Then _
                destText &= Mid(origText, counter, 1)
        Next counter
        Return destText
    End Function
```

الدالتين الأخيرتين، CountSubStrوGetSubStr،تحسبان وتستخرجان نصوص فرعية substrings من نصوص أكبر بالاعتماد على محددdelimiter . تتضمن الفيجوال بيسك دالتين، Mid و GetChar،تستخرجان أيضاً نصوص فرعية من نصوص أكبر،ولكنهما يعتمدان على موضع النص الفرعي.تتفحص الدالتين CountSubStr وGetSubStrالنصوص الفرعية أولاً باستخدام المحدد delimiter لتقسيم النص الأكبر في أجزاء.

```
Public Function CountSubStr(ByVal mainText As String, ByVal subText As String) As Integer
        ' العودة بإحصاء عدد المرات التي تم مصادفة وجود جزء نصي في نص رئيسي
        Dim totalTimes As Integer
        Dim startPos As Integer
        Dim foundPos As Integer
        totalTimes = 0
        startPos = 1
        ' مواصلة البحث حتى لم نعد نجد النص الجزئي أبدا
        Do
            ' البحث عن النص الجزئي
            foundPos = InStr(startPos, mainText, subText)
            If (foundPos = 0) Then Exit Do
            totalTimes += 1
            ' الانتقال لما بعد إيجاد النص
            startPos = foundPos + Len(subText)
        Loop
        ' إعادة الاحصاء
        Return totalTimes
    End Function
    Public Function GetSubStr(ByVal origString As String, ByVal delim As String, ByVal whichField As
Integer) As String
        ' استخراج نص محدد بمحدد معين من نص أكبر
        Dim stringParts() As String
        ' معالجة بعض الأخطاء.
        If (whichField < 0) Then Return ""
        If (Len(origString) < 1) Then Return ""
        If (Len(delim) = 0) Then Return ""
        ' تقسيم النص الأصلي إلى أجزاء محددة بمحدد
        stringParts = Split(origString, delim)
        ' لنرى هل الجزء الذي نريده موجود،فإذا كان كذلك لنرجع به
        If (whichField > UBound(stringParts) + 1) Then Return "" Else Return stringParts(whichField - 1)
    End Function
```

تحسب الدالة CountSubStr عدد مرات ظهور نص فرعي في نص أكبر.وتستخدم دالة الفيجوال بيسك InStr لإيجاد موضع النص الجزئي(subText) في نص أكبر(mainText).وتحافظ على عمل هذا حتى الوصول إلى نهاية mainText،والمحافظة على تشغيل الحساب(totalTimes) لعدد التطابقات.
ولتكون على دراية أكثر،قمت باستخدام مقاربة مختلفة لتنفيذ الدالةGetSubStr. وهذه الدالة تعود بمقطع محدد من نص ما.على سبيل المثال،العبارة التالية تحصل على comma-delimited portion الجزء المحدد بالفاصلة الثالث للنص bigString.

```
bigString = "abc,def,ghi,jkl,mno"
MsgBox(GetSubStr(bigString, ",", 3)) ' Displays: ghi
```

استخدمت دالة الفيجوال بيسك Splitلتقسيم النص الأصلي(origString)لعدة مصفوفات أصغر من النصوص(stringParts)،باستخدام delimكنقطة تقسيم.ومن ثم عملت على إرجاع رقم العنصر whichFieldمن النتيجة.بما أن whichFieldيبدأ بـ1 والمصفوفة تبدأ بـ0،فيجب علي ضبط الموضع للعودة بالعنصر الصحيح.

نماذج ويندوز Windows Forms

تقنية الدوت نت المعروفة بنماذج ويندوز *Windows Forms* تتضمن كل الفئات والميزات المطلوبة لتطوير تطبيقات "سطح مكتب"قياسية لميكروسوفت ويندوز.ولقد أصبحت الآن واحد من أنواع التطبيقات المتاحة على طول مع تطبيقات الكونسول،تطبيقات الانترنت ،والتطبيقات الخدمية ...إلخ.

الرسائل ومضخة الرسائل Messages and the Message Pump

عندما تتفاعل مع ويندوز فمن السهل عليك(كإنسان)أن تلاحظ النماذج المختلفة والأدوات التي على الشاشة.فبإمكانك التفاعل مع هذه النوافذ من خلال لوحة المفاتيح والماوس،يحتفظ ويندوز بقائمة النوافذ المعروضة على الشاشة،كيف تتداخل ويغطي بعضها الآخر،ولأي تطبيق تنتمي كل نافذة.(بعض التطبيقات تكون مجزأة إلى عدة مسارات threads وجميعها تعمل في نفس الوقت.في مثل هذه البرامج يحتفظ ويندوز بمسار جميع النوافذ على قاعدة كل مسار،وليس على قاعدة كل تطبيق فقط). يحدث كل إجراء مدخل من قبل المستخدم(مثل نقرات الماوس وضغطات المفاتيح)في طابور رسائل النظام *system message queue* بواسطة مشغل(أو معرف driver) الجهاز ذو الصلة.فعندما تنقر على الشاشة بواسطة الماوس يستخرج ويندوز رسالة النظام من هذا الطابور،محدداً أين نقرت،ومحاولاً استكشاف أي نافذة حدث نقر الفارة عليها،ومن ثم إعلام تطبيق ويندوز حول نقر الفارة بإضافة رسالة إلى طابور رسائل التطبيق.ويعمل نفس الشيء من أجل إدخال لوحة المفاتيح والإجراءات الأخرى التي ومن المحتمل يحتاج ويندوز أن يعرف شيء ما حولها.

لعمل وظيفة ضمن بيئة ويندوز،يتضمن تطبيقك(أو مسار معين ضمن تطبيقك)مضخة الرسائل *message pump*،وهي مقطع من الكود يراقب طابور الرسائل.تتضمن كل رسالة قادمة(أو داخلة) رقم المعرف ID للنافذة المقصودة.يستخرج الكود الرسالة من الطابور،ويمررها إلى إجراء النافذة(ويدعى أيضاً *WndProc*)للنافذة المالكة من أجل المعالجة النهائية.في لغة C تبدو مضخة الرسائل هذه مشابهة للتالي:

```
while (!done)
{
/* ----- Extract and examine the next message. */
MSG msg;
if (GetMessage(&msg, NULL, 0, 0))
  {
/* ----- WM_QUIT means it's time to exit the program. */
if (msg.message == WM_QUIT)
done = true;
/* ----- Send the message to the right window. */
TranslateMessage(&msg);
DispatchMessage(&msg);
  }
}
```

بالتالي كل تطبيق(عملياً،كل مسار ضمن تطبيق)لديه مضخة رسائل واحدة وإجراءات متعددة لكل نافذة.توجد مضخة الرسائل لتمرير أو إرسال الرسائل إلى إجراء النافذة الصحيح.

إجراءات النافذة.Window Procedures

بما أن مضخة الرسائل ترسل الرسائل إلى إجراءات النافذة المستقلة،يوجه روتين إجراء النافذة WndProc المعالجة إلى مقاطع الكود المستقلة أو الإجراءات المعتمدة على نوع الرسالة القادمة.إليك منطق عام(شبه كود) يوضح تركيب إجراء نافذة نموذجي:

```
If (message type is a mouse click)
Do mouse-click related code
Else If (message type is a key press)
Do key-press related code
Else If (message type is a window resize)
Do window-resizing-related code
Else...
```

(يستخدم شبه الكود عبارات If المتتالية،ولكن عملياً يستخدم إجراء نافذة بشكل أكثر شيوعاً عبارة "اختر حالة Select Case" "المعالجة الرسالة القادمة).لذلك ،يشبه إجراء النافذة آلة البيع،إذا ضغط الزبون زر الكولا،تقوم بمعالجة ما يعيد بعلبة كولا.وإذا ضغط الزبون على زر العلكة،تقوم بمعالجة ما يعيد علكة.وإذا ضغط الزبون على زر إعادة العملة المعدنية،تحتفظ بالنقود.

من أجل كل نوع رسالة(على الأقل بالنسبة لتلك التي يريد البرنامج معالجتها)،تتم معالجة بعض الكود عندما تصل الرسالة.فإذا عدنا إلى الفيجوال بيسك1.0،جميع التطبيقات المولدة تتضمن مضخة رسالة message pump وإجراءات WndProc من أجل كل نافذة،وجميعه مخفي.المهمة الرئيسية لإجراءات WndProc كان استدعاء الكود في معالجات حدث الفيجوال بيسك.

النوافذ في الدوت نت.Windows in .NET

خذها من شخص اعتاد كتابة تطبيقات ويندوز في لغة سيC :كتابة مضخات الرسائل message pumps وإجراءات النافذة window procedures ليس بتلك السهولة.عملت ميكروسوفت على محاولة تقنيع بعض الملل من خلال تقنيات متنوعة،من ضمنها فاحص الرسالة Message Crackers وفئات إطار العمل MFC. وفي النهاية نجحت فيجوال بيسك في دفن التعقيد تحت النظام المنطقي القريب من المبرمج programmer-friendly logical system .

يستخدم إطار عمل الدوت نت NET Framework. نظام مشابه تماماً للتنفيذات الأقدم في الفيجوال بيسك،باحتواء إجراء النافذة WndProc استدعاء معالجات حدث خاصة مكتوبة من قبلك.فهو يحزم جميع الفعالية والبساطة في تقنية تدعى نماذج ويندوز Windows Forms.جميع فئاتها تظهر في فضاء الأسماء System.Windows.Forms.العديد من هذه الفئات تنفذ أنواع معينه من النوافذ،مثل النافذة الرئيسية الاعتيادية،الأزرار،صناديق النصوص،قوائم الصندوق المركب المنسدلة،وهكذا.

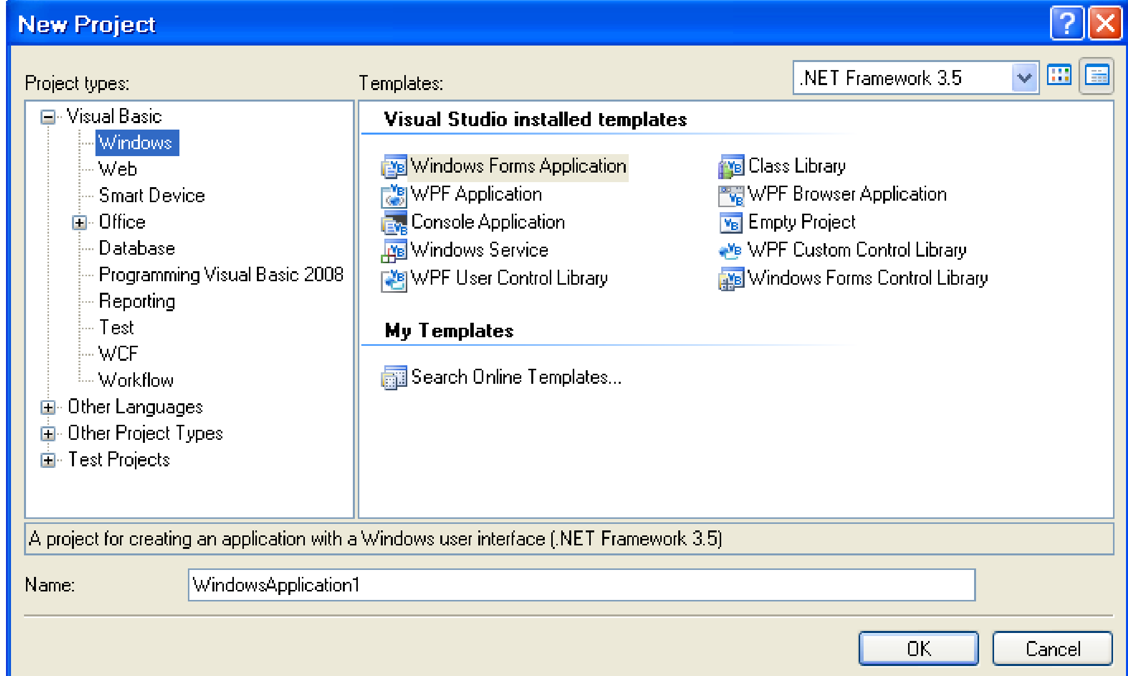
إذا كنت تريد،مايزال بإمكانك الوصول إلى مضخة الرسائل message pump وروتينات إجراء النافذة WndProcالمتنوعة.تتضمن كل فئة خاصة بنافذة طريقة WndProcتستطيع إعادة قيادتها وصنعها بنفسك.فمضخة الرسائل موجودة في الطريقة System.Windows.Forms.Application.Run.تستطيع أن تأمر أي من هذه المكونات والتحكم بكل شيء بنفسك،ولكنك ستكتشف أن معالج نماذج ويندوز ممتع جداً،وستعمل بجد لتنسى حتى ما هي مضخة الرسائل message pump أو ما تعني.

النماذج والأدوات Forms and Controls

في الدوت نت وكما في إصدارات فيجوال بيسك الأقدم،النوافذ تكون منضدة ضمن النماذج والأدوات.ولكنها ماتزال نوافذ،مبنية من نفس المكونات الجوهرية(القاعدية)،إذا كنت لا تصدقني،راجع فئات النماذج المتنوعة والأدوات،ففي الدوت نت كل من النماذج والأدوات تشتق من الفئة المشتركة System.Windows.Forms.Control،والتي تلخص التخصص الوظيفي الجوهري للنوافذ(نافذةwindow).بعض الأدوات المزودة في الدوت نت(وأيضاً مع فيجوال بيسك الأقدم)لاتنفذ عملياً على عناصر نافذة الشاشة،وهذه الأدوات مثل أداة "المؤقتTimer" فهو لايتضمن على واجهة ملموسة،ولكنها توفر برمجة حسية مشابهة للأدوات المرئية.

تصميم تطبيقات نماذج ويندوز Designing Windows Forms Applications

إنشاء تطبيق نماذج ويندوز Windows Forms application في الفيجوال بيسك سهل ،لنجربه،ابدأ فيجوال أستوديو واختر مشروع جديد New Project من قائمة ملف Fileويظهر نموذج المشروع الجديد كما هو مبين في الشكل التالي.



اختر نوع المشروع Project type "ويندوز Windows"،ومن ثم من نافذة القوالب Templates اختر قالب تطبيق نماذج ويندوز Windows Forms Application.امنح المشروع الاسم الذي تريد في حقل الاسمName، ومن ثم انقر موافقOK. سيكون للمشروع الجديد فورم(نموذجForm1)وحيد جاهز لتستخدمه.عند هذه النقطة،تكون الفيجوال أستوديو قد أضافت حوالي250 سطر من الكود المصدري لتطبيقك.إذا نقرت على زر إظهار جميع الملفات Show All Files في لوحة مستكشف الحلولSolution Explorer (تم شرحها سابقا في الفصل الأول)وافتح الملفات المتنوعة في المشروع،بإمكانك رؤية الكود بنفسك.أهم ما في الكود في الملف*Form1.Designer.vb*، تم تحديثه قليلاً هنا:

```
Partial Class Form1
    Inherits System.Windows.Forms.Form

    'Form overrides dispose to clean up the component list.
    <System.Diagnostics.DebuggerNonUserCode()> _
    Protected Overrides Sub Dispose(ByVal disposing As Boolean)
        Try
            If disposing AndAlso components IsNot Nothing Then
                components.Dispose()
            End If
        Finally
            MyBase.Dispose(disposing)
        End Try
    End Sub

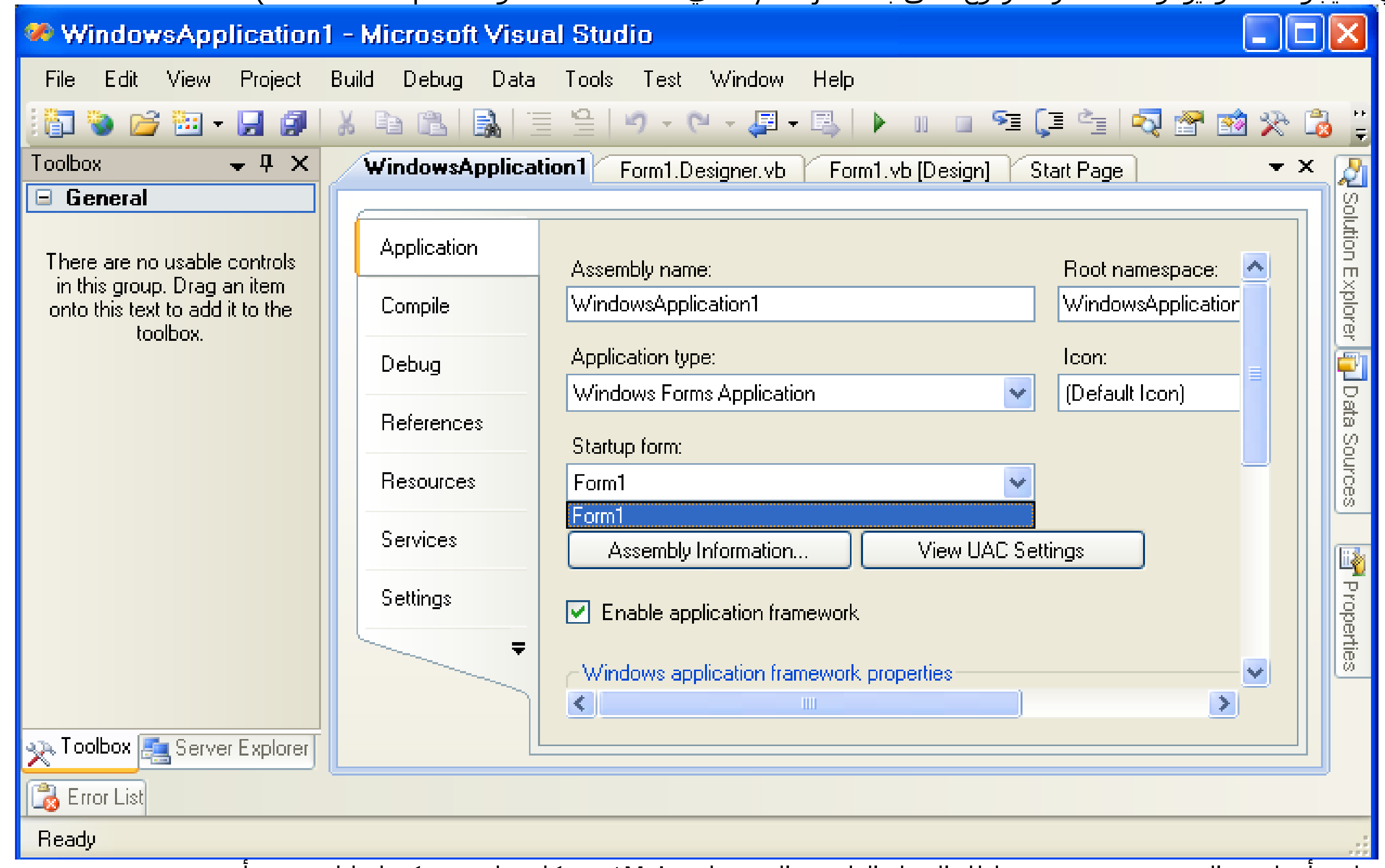
    'Required by the Windows Form Designer
    Private components As System.ComponentModel.IContainer

    'NOTE: The following procedure is required by the Windows Form Designer
    'It can be modified using the Windows Form Designer.
```

```
    'Do not modify it using the code editor.
    <System.Diagnostics.DebuggerStepThrough()> _
    Private Sub InitializeComponent()
        components = New System.ComponentModel.Container()
        Me.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font
        Me.Text = "Form1"
    End Sub

End Class
```

كل الكود الذي ينفذ سلوك النموذج يظهر في فئة النموذج(الفورم Form class) في فضاء الأسماءSystem.Windows.Forms. فورم المشروع الأولية، Form1، ترث من الفئة القاعدية base form، وتستقبل كل التخصص الوظيفي للفورم والإعدادات الافتراضية.وأي تغيرات وقت التصميم يتم عملها لواجهة المستخدم Form1،مثل إضافة أدوات أبناء،يتم إضافتها إلى إجراء "التمهيد للمكونات InitializeComponent"بشكل آلي عندما تستخدم الفيجوال أستوديو.
راجع هذا الروتين بشكل دوري periodicallyلرؤية كيف يتغير هذا الإجراء.
معظم البرامج سيكون لديها عدة فورمات،افترض أن بإمكان الدوت نت اختيار واحد من النماذج بشكل عشوائي لعرض متى يشتغل البرنامج للمرة الأولى. وهذا سيكون لهو وغير قابل للتنبؤ به unpredictable .إنك تشير إلى نموذج بداية التشغيل startingمن خلال خصائص المشروعproject's properties، بواسطة الحقل"نموذج الانطلاقStartup form " على تبويب التطبيق Application (شاهد الشكل التالي)(تظهر نافذة خصائص المشروع project's properties عندما تختار مشروع Project <<أمر القائمة خصائص Properties في الفيجوال أستوديو،أو عند النقر المزدوج على بند My Project في مستكشف الحلولSolution Explorer .)



عندما يبدأ تطبيق الدوت نت،يستدعي إطار العمل الطريقة المسماة Mainفي مكان ما ضمن كودك.إنك تشير أي نموذج رئيسي يتم استخدامه من خلال حقل " Startup form ". إن هذا الحقل يتضمن قائمة بكل النماذج ،ما عليك هو اختيار ما تريد منها.ولكن أنتظر،إنك لم تضيف الطريقة "رئيسية Main" إلى أي فورم؟لامشكلة،إن الفيجوال بيسك ستكتب روتين(إجراءMain routine) رئيسي بسيط دون تدخل منك والذي سيعرض النموذج المشار إليه.هذا الروتين يتم إضافته وقت الترجمة،فيعمل فقط المعالجة الأقل المطلوبة لعرض الفورم.
إذا أردت إضافة روتين رئيسي مخصص لفورمك،أو إلى بعض الفئات الأخرى غير النماذج في تطبيقك،فهذا ليس بالمشكلة،إذا كنت لم تختار(لم تضع علامة صح) على الحقل "Enable application framework" المبين في الشكل السابق،على نفس تبويب الخصائص المبين(لاحظ أنني قمت باختياره(وضعت إشارة صح فيه))القائمة "نموذج الانطلاقStartup form "تتغير لتضمين أي فئة في تطبيقك فيها روتين رئيسي مناسب(موافقcompatible).ولكن تطبيق إطار العمل يمكن الكثير من التخصص الوظيفي المقدر باعتدال cool functionality،وهذا كله بدون تدخل منك.عطل هذا الخيار فقط عندما تحتاج تحكم دقيق على عمر التطبيق منذ البداية(مبكرا).

استخدام طرق Mainمخصصة Using Custom Main Methods

إذا قررت كتابة روتين "رئيسي Main " خاص بك في فئة غير الفورم،فإنك تريد في النهاية عرض نموذج تطبيقك الرئيسي.بالعودة إلى الفيجوال بيسك6، فعندما تريد عرض نموذج ما،فإنك تستدعي الطريقة "أظهرShow ":

Form1.Show

هذا التركيب البسيط اختفى عندما ظهر الإصدار الأول للدوت نت من الفيجوال بيسك في2002،ولكن عاد مع الإصدار2005.قل إنك تريد بداية تطبيقك من الإجراء Sub Main في وحدة برمجية moduleمفصولة عن الفورم الرئيسي main form .أولاً،تحتاج لإضافة وحدة برمجية جديدة moduleإلى المشروع.اختر مشروعProject >>إضافة وحدة برمجيةAdd Module . عدل كود الوحدة البرمجية الجديدة بحيث يبدو مشابهة لمقطع الكود التالي:

```
Module Module1
    Public Sub Main()
        Form1.Show()
    End Sub
End Module
```

في نافذة خصائص المشروع ،لا تضع علامة صح لحقل" Enable application framework "،واختر إما Module1أو Sub Main من قائمة Startup form.وهذا بسيط جداً.إذا شغلت البرنامج الآن،سيعمل البرنامج وينغلق بسرعة.في الحقيقة،ستظهر Form1فقط لحظة مقتضبة قبل وجود البرنامج.لماذا لم تثبت ال Form1 ؟يتواجد البرنامج مباشرةً لأن مضخة الرسائل المزعجة،أو بدقة أكثر،العوز لمضخة الرسائلmessage pump. كل نافذة(أو فورم أو أداة)لديها إجراء "إجراء ويندوز WndProc "،ولكن يوجد فقط مضخة رسائل واحدة مميزة message pump لكل تطبيق أو مسار تنفيذthread. في هذا البرنامج البسيط، Form1 لديها إجراء "إجراء ويندوز WndProc " الخاص بها،ولكنه لايتحكم بمضخة الرسائل بنفسه.عليك على وجه التخصيص إخبار البرنامج ببدء تشغيل مضخة الرسائل message pump.بما أن مضخة الرسائل القياسية standard message pump لتطبيق نماذج ويندوز تظهر في الطريقة System.Windows.Forms.Application.Run،عدل كود Sub Main ليتضمنها مما يمكّن المضخة ويحفظ الفورم1 معروضة حتى تغلقها:

```
Module Module1
    Public Sub Main()
        System.Windows.Forms.Application.Run(Form1)
    End Sub
End Module
```

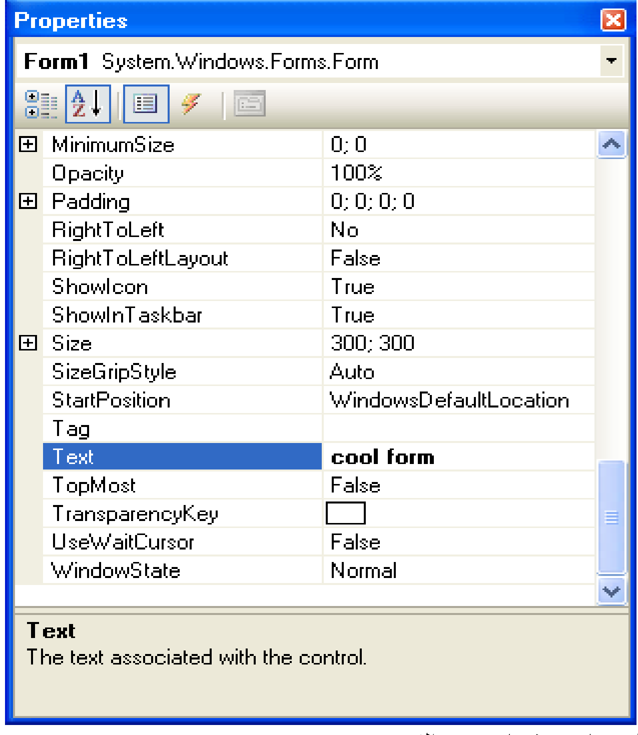
بإمكانك إضافة جميع الأنواع للكود الممهد لإجراءSub Main، وإظهار النموذج الرئيسي main form فقط عندما يكون كودك جاهز للتفاعل مع المستخدم.

إذا أردت إضافة نموذج جديد لتطبيقك،استخدم مشروع Project >> إضافة نموذج ويندوزAdd Windows Form .

العمل مع النماذج Working with Forms

في الدوت نت كل النماذج هي فئات بسيطة،تنوع عن الفئة System.Windows.Forms.Form.كل مرة تعمل على إنشاء نموذج جديد،فإنك تنشئ فئة مشتقة معتمدة على فئة النموذج المشتركةcommon Form class . ويتم تحميل فئتك الجديدة مع التخصص الوظيفي،وهي تتضمن الحقولfields، الطرقmethods،الأخطاءbugs ، الخاصيات properties،والأحداث eventsالتي تركب فئة الفورم.تأخذ الفيجوال أستوديو هذه العناصر وتجلبهم بطريقة تجعل منها سهلة البرمجة بالنسبة للفورم من خلال كل الكود المصدري source code ومن خلال السحب والإسقاط drag-and-dropعلى واجهة المستخدم لمصمم نماذج الفيجوال أستوديو Visual Studio Forms Designer.

عندما تضيف نموذج formللمرة الأولى لتطبيقك،استخدم لوحة الخصائص Properties (شاهد الشكل التالي)لتعديل الفورم لما تحب.وهذه اللوحة تظهر الخاصيات المبدئية للبنود المختارة حالياً ضمن بيئة الفيجوال أستوديو.فهي تتضمن مدخلة مفصولة لإعداد كل خاصية،معظمها يمكن أن يتم تحديثه من خلال مدخلة النص textالبسيطة.على سبيل المثال،بإمكانك تعديل العنوان المعروض في أعلى الفورم بتبديل المحتوى لخاصية النص Textمن Form1إلى Cool Form.



إليك بعض الخاصيات التي تظهر في نافذة الخصائص فيما يخص الفورم.

Propertyالخاصية	Descriptionالوصف
(Name)	هذه هي اسم الفورم،أو بدقة أكثر اسم الفئة التي تمثل هذه الفورم.يتم تسميتها بشكل افتراضي Form*x* لتستخدم في الكود حيث *x* هو عدد ما.وقد يكون من المتطلب تغير هذا الاسم لشيء ما أكثر دلالة.
AcceptButton	تشير لأي زر تم وضعه على الفورم سابقاً وسوف يطلق عند الضغط على مفتاح Enter.
AutoScroll	إذا وضعت هذا الحقل إلى صواب،فإن النموذج يضيف بشكل آلي أشرطة تمرير والتي تتحرك حول محتوى الفورم إذا ما تم تحجيم الفورم لتصبح صغيرة جداً لإظهار كل شيء.
BackColor	لون خلفية النموذج.
BackgroundImage	استخدم هذه الخاصية مع الخاصية BackgroundImageLayout لوضع صورة على خلفية الفورم.
CancelButton	تشبه الخاصية AcceptButton،ولكن لإسناد زر يتم إطلاقه عند الضغط على المفتاح Esc.
ContextMenuStrip	تسمح لك من إنشاء قائمة اختصار مخصصة تظهر عندما ينقر المستخدم يمين على خلفية الفورم.تشير ContextMenuStrip إلى أدوات مفصولة تعمل على إضافتها للنموذج.
ControlBox	إظهار ورؤية صندوق التحكم في الزاوية اليسارية العلوية للفورم من خلال إعداد هذه الخاصية.
Cursor	تشير إلى تخطيط مؤشر الفارة.
FormBorderStyle	هذه الخاصية تشير إلى نوع الفورم المعروضة.الافتراضي قابلة للتحجيم Sizable.
Icon	تضع صورة معروضة في الزاوية العلوية اليسارية لحدود الفورم.
IsMdiContainer	تمكن دعم"واجهة متعددة مستندات multiple document interface" على هذه الفورم.وهذا ما يتيح للنموذج الرئيسي أن يحتوي عدة نماذج أبناء. يمكن أن تعرض الفيجوال أستوديو نفسها والنماذج ونافذة الكود المصدري في تخطيط بيئة متعددة المستندات MDI style.
KeyPreview	إذا وضعت هذه الخاصية لصواب،فإن أحداث الفورم KeyDown وKeyPress ستحصل على معالجة أي مفتاح تم إدخاله من قبل المستخدم،حتى ولو كانت هذه المفاتيح القصد منها التحكم بالمحتوى على الفورم.وهذا مفيد عند الحاجة لالتقاط المفاتيح المقدمة لكامل الفورم،مثل استخدام المفتاح F1 لإطلاق مساعدة عبر الانترنت.
Location	موقع الفورم على الشاشة وتوثر أيضاً الخاصية StartPosition في موقع الفورم.
MainMenuStrip	تحدد أداة عروة القائمة" MenuStrip لاستخدام القائمة الرئيسية للفورم.يتم إضافة أداة عروة القائمة MenuStrip المشار إليها بشكل منفصل للنموذج.
MaximizeBox	ظهور أو عدم ظهور صندوق التكبير في الزاوية اليسارية العليا.
MinimizeBox	ظهور أو عدم ظهور صندوق التصغير في الزاوية اليسارية العليا.
MinimumSize	إمكانية تحجيم الفورم.(الحد الأصغري لتحجيم الفورم).
Opacity	الشفافية،تسمح بتخصيص مستوى الشفافية للون معين
ShowInTaskbar	تخصص فيما إذا هذه الفورم يجب أن تظهر كبند في شريط مهام النظام.
Size	تشير إلى الحجم الحالي للفورم من خلال الخاصيات الفرعية العرض Width والارتفاع Height .
StartPosition	كيفية وضع الفورم على الشاشة عندما تظهر للمرة الأولى على الشاشة.

Tag بإمكانك وضع أي نوع بيانات تريدها في هذه الخاصية،فهي موجودة لكي تستخدمها.

لقد جدولت أقل من نصف الخاصيات هنا،فمن الواضح أن لديك الكثير من التحكم على الفورم وكيفية إظهارها للمستخدم.ما يهم فعلياً هو أن العديد من هذه الخاصيات غير مقصورة على النماذج.بعض من هذه الخاصيات تأتي من فئة System.Windows.Forms.Control المشتركةmutual،وتظهر أيضاً في كل الأدوات الأخرى التي تستخدم نفس الفئة القاعديةbase class. و تتضمن هذه الخاصيات مثل الموضعLocation، لون الخلفيةBackColor، والنصText، على الرغم من أن النص المعروض في عنوان الفورمcaption والنص Text المعروض على زر الأمر يختلفان بشكل ملحوظ في إيجادهم،فإن الاستخدام في الكود متطابق.

```
Form1.Text = "This is a form caption."
Button1.Text = "This is a button caption."
```

على الرغم من أنك تستطيع وضع كل الخاصيات من نافذة الخصائص،بإمكانك أيضاً تحديث وعرضها من خلال الكود.في الحقيقة،إذا ما عدلت أي من الخاصيات من خلال نافذة الخصائص،فتكون قد حدثتهم خلال الكود المصدري،بما أن الفيجوال أستوديو يعمل على تحديث كودك تماما دون تدخل منك.جرب هذا،ضع أي من الخاصيات في لوحة الخصائص للفورم،ومن ثم اعرض الإجراء "التمهيد للمكونات InitializeComponent" في ملف*Form1.Designer.vb*. تستطيع مثلا وضع الخاصية TopMostإلى صواب،وستجد العبارة الجديدة التالية قرب النهاية السفلية للإجراء.

```
Me.TopMost = True
```

بإمكانك عمل هذا من جانبك لتخصيص الخصائص من خلال الكود فما عليك سوى وضع الكائن متبوع بنقطة ومن الخاصية التي تريد كما فعل الفيجوال أستوديو مع الخاصية TopMost.

```
Me.Text = "The Library Project"
```

بإمكانك أيضاً استخراج قيمة الخاصية من خلال أسماءها كما يلي:

```
MsgBox("عنوان هذا الفورم: " & Me.Text)
```

بإمكانك الوصول لطرق الفورم المتنوعة بنفس الطريقة.على سبيل المثال،الطريقة إغلاق التي تغلق الفورم:

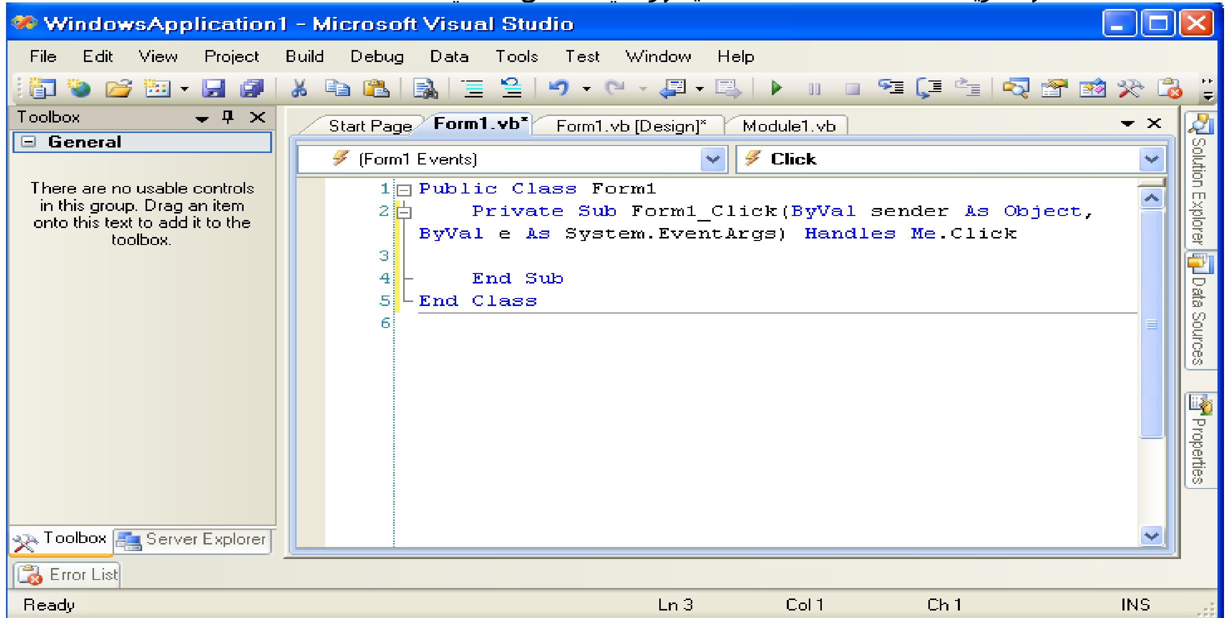
```
Me.Close()
```

بالطبع هذه العبارة تحتاج أن تظهر ضمن إجراء ما محقق مثل معالج حدث ما event handler ، لنضيف بعض الكود لحدث نقر Click event على الفورم لذلك عندما ينقر المستخدم على الفورم فإن الكود الجديد سيعمل على تبديل عنوان الفورم بحيث يذكرنا ما هو هذا العنوان ، وإغلاق الفورم يؤدي إلى الخروج من البرنامج . للوصول Accessإلى الكود المصدري اختر Form1.vb في مستكشف الحلول Solution Explorer ومن ثم انقر على زر عرض الكود View Code عند أعلى مستكشف الحلول . يظهر مقطع كود الفورم الافتراضي كمايلي .

```
Public Class Form1

End Class
```

هذه هي الحصة المتنقلة لفئة الفورم Form1، الجزء الذي تظهره الفيجوال استديو للعامة ، وليس الأجزاء المخفية الأكثر أهمية (الجزء في *Form1.Designer.vb*) . ولكن سنكون قادرين على جعل هذا المقطع هام . أضف حدث نقر Click eventلسطح النموذج form's surface باختيار (أحداث الفورم Form1 Events)من قائمة اسم الفئة Class Name (أعلى ويسار محرر نص الكود)، ومن ثم اختر نقر من القائمة المنسدلة لاسم الطريقة Method Name ، كما يظهر في الشكل التالي :



عدل معالج الحدث بحيث يعرض الكود المبين هنا:

```
Private Sub Form1_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Click
        Me.Text = "The Library Project"
        MsgBox("The form's caption is: " & Me.Text)
```

```
        Me.Close()
    End Sub
```

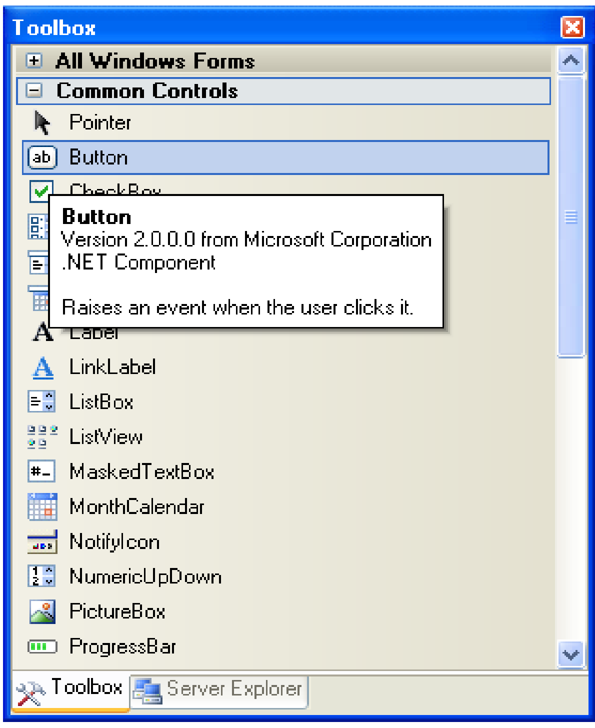
إذا شغلت التطبيق ونقرت على سطح الفورم،سيظهر صندوق رسالة مع عنوان الفورم قبل الخروج من التطبيق لاحظ الشكل التالي.

إضافة أدوات Adding Controls

تتضمن الفيجوال بيسك والدوت نت الكثير من أدوات نماذج ويندوز،وحتى يمكن عمل أدوات واستخدامها كالأدوات العادية الموفرة في صندوق الأدوات. بإمكانك بناء أدواتك الخاصة،إما باشتقاقها من فئات أدوات موجودة أو تنفيذهم بالكامل من الصفر.



صندوق أدوات Toolboxالفيجوال أستوديو يتضمن جميع الأدوات الأساسية التي تحتاجها لبناء تطبيقات برمجية عالية النوعية high-quality
أو حتى منخفضة النوعية وإجحافpathetic low-quality. للوصول إلى صندوق الأدوات-الجزء الذي يظهر في الشكل التالي-من خلال عرض View >> صندوق الأدواتToolbox .



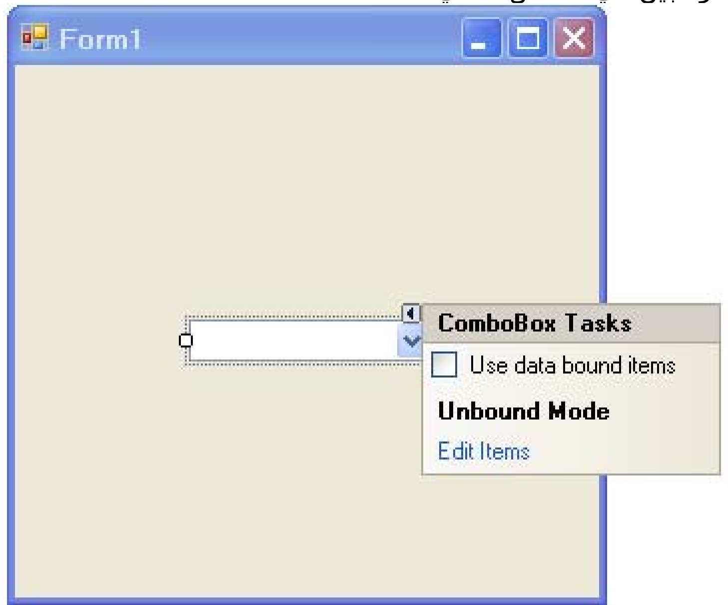
يوجد أربع طرق لإضافة أداة لفورم:
1.النقر المزدوج على الأداة في صندوق الأدوات.فتظهر نسخة من الأداة على سطح الفورم في موقعها الافتراضي مع جميع الإعدادات الافتراضية.
2.سحب وإسقاط الأداة من صندوق الأدوات Toolboxإلى الفورمform .

3.انقر على الأداة في صندوق الأدوات،ومن ثم استخدم الماوس لسحب المنطقة المستطيلة على الفورم حيث ستظهر الأداة.بعض الأدوات،مثل أداة الصندوق المركبComboBox ،لديها حدود بالنسبة للعرض والارتفاع،فليس من الضروري أن تحجم نفسها عند تنوي.
4.أضف أداة للفورم باستخدام الكود المصدري للفيجوال أستوديو.كما تضيف أدوات للفورم في الفيجوال أستوديو،فهي تكتب الكود المصدري لك.لايوجد سبب لعدم إضافة هذا الكود بنفسك،على الرغم من أنه يوجد تحذير في الملف*Form1.Designer.vb* يخبرك بعدم تحديث هذا الملف،بإمكانك تعديل الإجراء InitializeComponentsبشكل يدوي إذا ما طابقت بدقة نمط الكود المولد بواسطة الفيجوال أستوديو.بإمكانك أيضاً إضافة أدوات في مناطق أخرى من كودك،مثل حدث تحميل الفورم.إضافة أدوات بشكل آلي يقع فيما وراء مجال هذا الكتاب،ولكن تابع وجرب هذا.

بعض الأدوات ليس لديها واجهة مستخدم حقيقية موجودة في التطبيق المشغل.هذه الأدوات،عند إضافتها للفورم،تظهر في لوحة أسفل سطح الفورم.وتستطيع الاستمرار في التفاعل معها تماماً مثل الأدوات المعتمدة على الفورم.

حالما تظهر الأداة على الفورم،استخدم الماوس لنقل الأداة أو إعادة تحجيمها باستخدام المراسي(الأناكرanchors) التي تظهر عند يتم اختيار الأداة.العديد من الأدوات تكون محدودة في خيار التحجيم.أداة الصندوق المركبComboBox ،على سبيل المثال،يمكن فقط إعادة تحجيمها أفقياً،وحجمها الشاقولي يتم تحديده بواسطة الأشياء مثل الخط fontالمستخدم في الأداة.بينما تسمح لك أدوات أخرى إعادة تحجيمها.ولكن فقط في بعض الأحيان.أداة الرقعة(اللصاقةLabel)يمكن تحجيمها يدوياً عندما تكون خاصيتها AutoSizeقد تم وضعها إلى خطأ False فقط.

تتضمن بعض الأدوات زر سهم صغير،ويظهر على الأغلب قرب الزاوية العلوية اليمينية من الأداة.وهو عبارة عن بطاقة(اقتباس سريعSmart Tag)،مشابه لميزة الاقتباس السريع المضمنة في ميكروسوفت أوفيس.يوفر النقر على الاقتباس السريع الوصول السريع إلى الميزات المصاحبة للأداة.كما هو مبين في الشكل التالي.



تبين القائمة التالية بعض الأدوات الشائعة الأكثر استخداماً،وكلها مضمنة بشكل افتراضي مع تطبيق نماذج ويندوز Windows Forms application وبشكل افتراضي في صندوق الأدواتToolbox . إذا ما أنشئت تطبيق نماذج انترنت web-based applicationsوفي الفيجوال أستوديو-المستخدم لتصميم تطبيقات معتمدة على الويب باستخدام ASP.NET –فإن الأدوات المتاحة ستختلف عن هذه القائمة.

الأيقونةIcon	الأداةControl	الوصفDescription
	BackgroundWorker	تتضمن الدوت نت دعم للتطبيقات ذات مسارات التنفيذ المتعددة multithreaded.تتيح لك الأداة BackgroundWorker البدء بمهمة الخلفية background taskتماماً من الوضعية المريحة لنموذجك.وهي مفيدة بشكل خاص عندما ترغب في تحديث فورم بالاعتماد على عرض العناصر المتفاعلة مع مسارات تنفيذ"worker" أخرى.إنك تستأنف مهمة عمل جديدة من خلال طريقة الأداة RunWorkerAsync،وتنجز العمل الحقيقي في حدثها DoWork.
	Button	زر دفعي قياسي.إن حدث النقر Clickهو الميزة الأكثر استخداماً من الناحية البرمجية.على الرغم من أنك تستطيع استخدام الخاصية DialogResultلإطلاق إجراء حوار خاص.
	CheckBox	تنفذ هذه الأداة طريقتين(فعالon ،غير فعال off)أو ثلاث طرق(فعالon ،غير فعالoff ،أخرother) لحقل الاختيار"اختبارchecked ".تشير الخاصية ThreeStateإلى العدد الكلي من الخيارات.استخدم الخاصية المنطقية "مختبر Checked " من أجل checkboxes صناديق الاختبار ذات الطريقتين،أو الخاصية CheckStateمن أجل صناديق الاختبار ذات ثلاث طرق.
	CheckedListBox	تدمج هذه الأداة أفضل مافي كل من أداة صندوق القائمة ListBoxوصندوق الاختبار CheckBox ،تمنحك قائمة حيث يكون كل بند فيها يمكن أن يتم اختباره بطريقتين أو ثلاث.الطريقة GetItemCheckedوالطريقة GetItemCheckState(ونظيرهما "وضعSet ")توفران واحد من عدة طرق لتفحص حالة بنود في القائمة.كن على دراية أن أداة مشابهة مسماة CheckBoxList و هي من أجل الاستخدام في تطبيقات ASP.NET فقط.
	ColorDialog	تعرض نموذج ويندوز قياسي مستخدم لاختيار لون من قبل المستخدم.عرض حوار اللون باستخدام طريقة الأداة

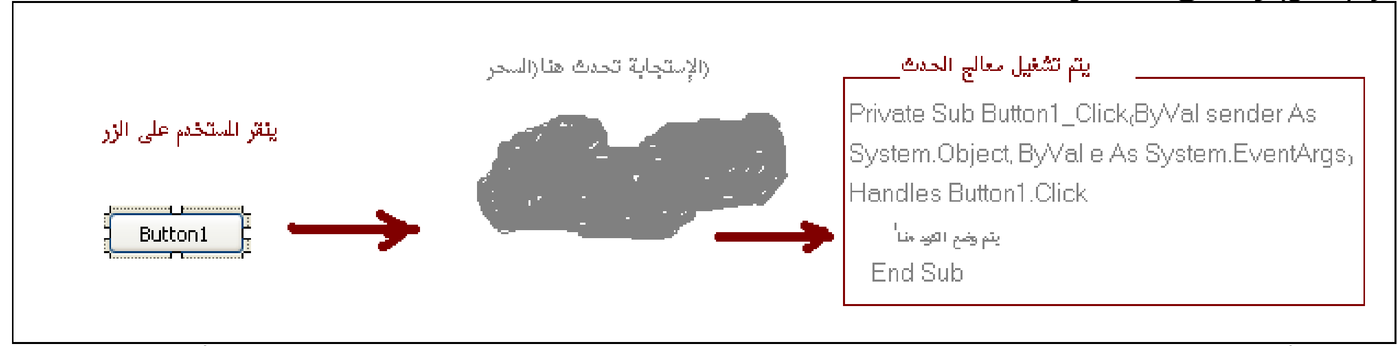
	ShowDialog،وتحصل على النتيجة بواسطة الخاصية Color.
ComboBox	تنفذ هذه الأداة صندوق مركب ComboBox منسدل للأسفل لنافذة قياسية،وفي معظم أنماطها المتنوعة. يمكن للقائمة أن تتضمن أي كائن ترغب به،فهي غير محصورة فقط بالسلاسل الحرفيةstrings. وتستطيع أيضاً توفير كود "ownerdraw"مخصص يسمح لك رسم كل بند قائمة بنفسك.
ContextMenuStrip	هذه الأداة تتيح لك تصميم اختصار أو قائمة"سياقcontext"،ليتم عرضها عندما ينقر المستخدم يمين على الفورم أو أداة تختارها أنت.ويتم تصميمها واستخدامها في الكثير من الشبه كما هو الحال لأداة MenuStrip.
DataGridView	تنفذ هذه الأداة جدول قياسي مثل الشبكة المستخدمة لعرض أو تحديث بيانات في خلايا مستقلة.ويتم تحميلها مع عرض خيارات كثيرة.يمكن للبيانات المعروضة أن ترتبط بمصدر بيانات خارجي،أو بإمكانك تركيبها على الطاير النمط."virtual data" يتيح لك أيضاً تحميل بيانات عند الحاجة فقط.
DateTimePicker	تتيح هذه الأداة للمستخدم إدخال تاريخ،وقت أو كلاهما،من خلال إما مدخلة النص الأساسية أو الأدوات المعتمدة على الماوس.على الرغم من أنه نموذج غير حر كحقل النص البسيط،فهو يعمل على إجبار اختيار لتاريخ ما أو وقت ما.بإمكانك وضع النهايات العظمى والصغرى على اختيار المستخدم.توفر الأداة MonthCalendarواجهة بديلة لاختيار تاريخ مخصص.
DomainUpDown	من خلال هذه الأداة،يختار المستخدم نموذج من بين قائمة من الخيارات التي تعرفها،الخيارات التي لها ترتيب ملزم خاص. استخدم هذه الأداة كبديل عن أداة ComboBoxأو TrackBarعندما يكون هناك مسوغwarranted .
FolderBrowserDialog	تعرض نموذج ويندوز القياسي المستخدم من أجل اختيار دليل أو مجلد من قبل المستخدم.تعرض صندوق حوار الاختيار باستخدام طريقة الأداةShowDialog، والحصول على النتيجة باستخدام الخاصية SelectedPath.
FontDialog	تعرض نموذج ويندوز قياسي يتم استخدامه من أجل اختيار الخط من قبل المستخدم.وتعرض صندوق حوار الاختيار باستخدام الطريقة ShowDialog،والحصول على النتيجة بواسطة الخاصيةFont. وهناك خاصيات أخرى تمكن الوصول إلى مكونات الخط المختار
GroupBox	توفر طريقة بسيطة لتجميع الأدوات بشكل مرئي على الفورم.الأدوات التابعة يتم رسمها أو لصقها مباشرة على هذه الأداة. للوصول إلى نفس التخصص الوظيفي بدون حدود مرئية أو عنوان استخدم الأداةPanel .
HelpProvider	تتيح لك الإشارة إلى تفاصيل المساعدة عبر الشبكة من أجل أدوات أخرى على الفورم.عند استخدامها فإنها تضيف العديد من شبه الخاصيات "المساعدةHelp "لكل أداة فورم أخرى من خلالها تستطيع تزويد تفاصيل سياق المساعدة.عند تنفيذها بشكل مناسب،سيتم عرض محتوى المساعدة عبر الشبكة عندما يضغط المستخدم على المفتاح F1في سياق الأداة الفعالة.
HScrollBar	تنفذ هذه الأداة شريط التمرير الأفقي،بحيث تسمح للمستخدم من التمرير خلال قطاع العرض أو قائمة من الاختيارات.من أجل التنفيذ الشاقولي لهذه الأداة،استخدم الأداةVScrollBar. تتضمن العديد من الأدوات الأخرى نسخة خاصة بها لأشرطة التمرير هذه.
ImageList	تغلف هذه الأداة مجموعة من الصور الصغيرة أو الأيقونات من أجل الاستخدام من قبل أدوات أخرى تدعم قائمة صور. يتم استخدام قوائم الصور بشكل شائع من قبل أداة عرض القائمةListView، شريط الأدوات Toolbar،وأداة عرض الشجرة TreeView.
Label	تعرض هذه الأداة نص ثابت على الفورم.وتستطيع رسم خطوط مستطيلات وما إلى ذلك.
LinkLabel	تنفذ هذه الأداة كما في الأداة السابقة ولكنها تتضمن وصلة أو أكثر links ضمن محتوى النص.وهي مشابهة لنص الوصلات القياسي في محتوى مستعرض الويب،تستدعي معالج حدثها LinkClickedعندما ينقر المستخدم على أي من الوصلات المضمنة.
ListBox	تنفذ هذه الأداة أداة صندوق نص قياسي،عارضة قائمة من البنود بحيث يستطيع المستخدم اختيار ما يشاء.يمكن للبنود المضمنة أن تكون أي كائن ترغب،فهي ليست محدودة لنصوص فقط.وتستطيع توفير كود حامل رسم ownerdraw يسمح لك رسم كل بند قائمة بنفسك.
ListView	تحضر هذه الأداة مجموعة من البنود مع خاصيات عرض اختيارية.تستطيع إضافة بيانات خاصة بعمود من أجل عرض التفاصيلdetails. تظهر البنود في الأداة كمجموعة من كائنات الفئة ListViewItem.
MaskedTextBox	تنوع عن صندوق النص القياسي يساعد المستخدم على إدخال عدد منسق أو بيانات نصية بعرض قالب أو قناع،على سبيل المثال تستطيع إجبار المستخدم على إدخال رقم الهاتف في هيئة"xxx-xxxxxx" باستخدام القناع العددي مع تضمين رموز الشحطة أو الواصلة القصيرة.
MenuStrip	تتيح لك هذه الأداة تصميم قوائم الفورم القياسية،والتي يتم عرضها على طول أعلى الفورم.يتم تنفيذ القوائم ضمن شريط القائمة من خلال حالات(نسخ) الفئةToolStripMenuItem. وشريط القائمة هو شريط أدوات يشبه في تنفيذه قائمة ويندوز القياسية.تستطيع إضافة أنواع أخرى من الأدوات للقائمة،متضمنة شريط أدوات خاص بأدوات الصندوق المركب.قوائم حساسة للسياق،يتم عرضها عندما ينقر المستخدم على الفورم أو أداة معينة،ويتم تنفيذها من خلال الأداة ContextMenuStrip.
MonthCalendar	تعرض مجموعة جزئية من التقويم،مركزة العرض على العرض الخاص بالشهر.يمكن أن يتم عرض أكثر من شهر في نفس الوقت.في تركيب شاقولي ،أفقي،أو في تركيب شبكة.توفر الأداة DateTimePickerواجهة بديلة من أجل الاختيار الخاص بالتاريخ.
NotifyIcon	تتيح لك وضع أيقونة في منطقة"صينية النظامsystem tray " لشريط مهام ويندوز،وربط رسائل مهمة إلى المستخدم من خلال هذه الواجهة.بما أن هذه الأداة ليس لديها واجهة مستخدم خاصة بالفورم،من الممكن استخدامها بدون أن يكون لديها عرض نموذج قياسي.
NumericUpDown	تسمح للمستخدم من اختيار قيم عددية باستخدام طريقة مقطع أعلى/أسفل القابل للتمرير.استخدم هذه الأداة كبديل عن الأدوات HScrollBar، TextBox، TrackBar، أو VScrollBarعندما يكون هناك مسوغ.
OpenFileDialog	تعرض نموذج ويندوز القياسي المستخدم لفتح ملف مختار من قبل المستخدم.يستطيع المستخدم اختيار واحد أو أكثر من الملفات الموجودة من أنظمة الملفات المحلية أو البعيدة.تعرض صندوق حوار الاختيار باستخدام طريقة الأداةShowDialog، تحصل على النتيجة بواسطة الخاصية FileNameأو الخاصيةFileNames، توفر الطريقة OpenFileطريقة سريعة لفتح ملف مختار.

الأداة	الوصف
PageSetupDialog	تعرض نموذج ويندوز قياسي يتم استخدامه لتركيب صفحة الطباعة بواسطة المستخدم.تعرض صندوق حوار الاختيار بواسطة طريقة الأداةShowDialog ،تحصل على النتيجة بواسطة الخاصية PageSettings والخاصيةPrinterSettings.
Panel	هذه الأداة عبارة عن أداة تجميع على الفورم.الأدوات التابعة يتم رسمها أو لصقها على هذه الأداة،للوصول إلى نفس التخصص الوظيفي مع حدود ظاهرة وعرض عنوان للمستخدم استخدم الأداة GroupBox.
PictureBox	تعرض هذه الأداة صورة بهيئات متنوعة.
PrintDialog	تعرض نموذج ويندوز قياسي يتم استخدامه لاختيار خاصية الطباعة وطباعة مستند من قبل المستخدم.تعرض صندوق حوار الاختيار باستخدام طريقة الأداة ShowDialog.يتم استخدام هذه الأداة بالعلاقة مع حالة من الفئة System.Drawing.Printing.PrintDocument،والتي يتم إنشائها من خلال الكود أو بواسطة الأداة PrintDocument.
PrintDocument	يتم استخدام هذه الأداة كجزء من عمليات معاينة الطباعة والطباعة.فهي تضيف دثار حول تنفيذ الطباعة الخاصة بك،موفرة طريقة ثابتة لاختيار وطباعة صفحات مستند.
PrintPreviewDialog	توفر هذه الأداة واجهة قياسية من أجل معاينة الطباعة،منفذةً جميع العناصر لحوار معاينة الطباعة بالكامل. عندما تستخدمها مع الفئة أو الأداة PrintDocument،فهي تعرض بدقة على الشاشة ما سيظهر على صفحة الطباعة النهائية.في الحقيقة،ليس من الضروري أن يعلم كودك فيما إذا سيرسل طباعته إلى الطابعة أو عرض معاينة الطباعة.
ProgressBar	يوفر تغذية راجعة رسومية للمستخدم من أجل مجال إتمام مهمة.عادةً،يكون المجال بين 0 % و100%،ولكن تستطيع توفير مجال مخصص.تشير الخاصية Valueللإعدادات الحالية بين حدود المجال الأدنى Minimumوالأعلى Maximum.
PropertyGrid	تسمح هذه الأداة للمستخدم من تحرير أعضاء معينة لحالة فئة مرافقة بشكل ظاهر أو تصويري.ولوحة خاصيات بيئة الفيجوال أستوديو هي حالة عن هذه الأداة.تستخدم هذه الأداة بشكل كبير المواصفات المعمدة على فئة للتحكم بالعرض وتحرير ميزات الخاصيات.
RadioButton	تنفذ هذه الأداة زر اختيار تبديل radioويندوز قياسي.على الرغم من عرض نقطة دائرية أكثر شيوعاً،يمكن أن تظهر الأداة أيضاً كزر مفصلي toggle buttonبوضع الخاصية Appearanceبشكل مناسب.تشير الخاصية Checkedإلى القيمة الحالية للأداة.جميع أدوات زر التبديل RadioButtonالتي تظهر ضمن نفس "سياق المجموعة group context " تتصرف بأسلوب خاص مشترك،استخدم الأدوات Panel وGroupBoxلإنشاء سياقات لمجموعة معينة.
ReportViewer	تسمح لك هذه الأداة من تصميم وعرض تقارير محزمة ومربوطة إلى تجمعات أو مصدر بيانات آدو دوت نت ADO.NET.وتعمل أيضاً مع خدمات تقارير سكول سرفر SQL Server المولدة للتقارير. استخدام هذه الأداة لتصميم تقرير سيضيف ملف.*rdlc* إلى تطبيقك والذي يحتوي تصميم تقرير حقيقي.
SaveFileDialog	تعرض هذه الأداة نموذج ويندوز قياسي يتم استخدامه لاختيار حفظ save ملف من قبل المستخدم.يمكن للمستخدم من أن يختار ملف جديد أو موجود من أنظمة الملفات المحلية أو البعيدة.وتطلب هذه الأداة من المستخدم بشكل اختياري من معاينة الملفات الموجودة.عرض صندوق حوار الاختيار باستخدام الطريقة ShowDialog،والحصول على النتيجة بواسطة الخاصية FileName.توفر الطريقة OpenFileطريقة سريعة لفتح الملف المختار.
SplitContainer	تضيف هذه الأداة شريط تقسيم(أو تقطيعsplit ba)تستطيع بواسطته تقسيم فورمك إلى عدة قطاعات قابلة للتحجيم،كل منها يحتوي أداة لوحةPanel. استخدم الخاصية Orientationلتبديل اتجاه التقسيم.سيتأثر الترتيب الذي تضيف فيه أدوات SplitContainerبقابلية استخدام التقسيمات،،أوصي بالتدريب على هذه الأداة.
StatusStrip	تعرض هذه الأداة شريط الحالةstatus bar ،عادة على طول الحافة الدنيا من فورمك،من خلالها تستطيع عرض حالة ،ومعلومات حساسة بالسياق أخرى للمستخدم.يمكن للشريط من أن يحتوي عدة أدوات أشرطة تقدمProgressBar، لوحة شريط الحالةStatusStripPanel، وأداة عنوان شريط أدواتToolStripLabel .
TabControl	تتيح لك تقسيم الأدوات التي على فورمك إلى عدة قطاعات مجدولة.كل اسم محدد(مسمى)لديه أداة صفحة جدولةTabPage ، والتي تعمل بشكل مشابه كثيراً إلى . أضف أو الصق أدوات تابعة مباشرة إلى كل أداة TabPage.
TextBox	تنفذ هذه الأداة صندوق نص ويندوز قياسي،في نمط وحيد السطر singleline ومتعدد الأسطرmultiline . محتوى الجسم الرئيسي هو مجموعة على طول خاصية النصText . تسمح لك الخاصيات PasswordCharو UseSystemPasswordCharمن تقنيع المدخلات عندما يعمل المستخدم على كتابة كلمة المرور.
Timer	تطلق هذه الأداة حدث التوقيت بالفترة interval التي تحددها أنت.قياس الفترة،بالميلي ثانيةmilliseconds،يتم وضعها من خلال الخاصية Interval.إذا تم وضع الخاصية Enabledإلى الصواب True،سيتم استدعاء معالج الحدث Tickكل مرة يقابل(أو يلاقي)الفترة.على الرغم من أنك تستطيع وضع فترة صغير كميلي ثانية واحدة،فمن غير المحبب أن تعمل هذا التحقق بشكل متكرر مع أجهزة الكمبيوتر اليومية.
ToolStrip	تنفذ هذه الأداة شريط أدواتtoolba تظهر عليه أدوات أخرى.وتأتي مع مجموعة أدوات مرافقة وفئات توفر معالجة وميزات تفاعل المستخدم متقدمة.
ToolStripContainer	توفر طريقة مريحة لإضافة أدوات شريط قائمة MenuStrip،شريط حالةStatusStrip، وشريط أدوات ToolStripلحواف الفورم.
ToolTip	تتيح لك هذه الأداة تحديد "تلميح أداةtool tip "من أجل الأدوات الأخرى على الفورم.عند استخدامها،فهي تضيف شبه خاصية "تلميح أداةToolTip " لكل أدوات الفورم الأخرى،من خلالها تستطيع تزويد نص تلميح الأداة المرافق.عندما تحوم الفارة فوق أداة يظهر نص التلميح المرافق،وذلك من خلال ظهور نافذة نص صغيرة بشكل مؤقت فوق الأداة لتوفير معلومات مفيدة للمستخدم.
TrackBar	تسمح للمستخدم من عمل اختيار من بين عدد صغير من القيم المرتبة ذات الصلة.وهو موازن(أو مثيل أو نظير)للعالم الحقيقي لأداة التحكم بحجم الصوت على الراديو.استخدم هذه الأداة كبديل للأداة HScrollBar و NumericUpDown أو VScrollBar عندما يكون هناك مسوغ.
TreeView	تجلب هذه الأداة مجموعة بنود في ترتيب هرمي.وهي مشابهة تماماً إلى قسم"شجرة الأدلةdirectory tree " لمستكشف ملفات ويندوز أو فيستا.كل بند في الشجرة هو عقدة node يمكن أن يكون لديها صفر عقدة أو عقد أبناء أكثر.
VScrollBar	تنفذ هذه الأداة شريط تمرير شاقولي،مع السماح للمستخدم من التمرير شاقولياً على القطاع المعروض أو قائمة اختيارات.من أجل التنفيذ الأفقي لهذه الأداة، استخدم الأداةHScrollBar، تتضمن العديد من الأدوات الأخرى نسخة خاصة بها لأشرطة التمرير

		هذه .
	WebBrowser	تنفذ مستعرض ويب ضمن تطبيقك،تستطيع استخدام ميزات بحث معتمدة على ويب قياسي متاحة ضمن مستكشف الانترنت Internet Explorer من أجل الوصول المعتمد على عناوين الانترنتURL، أو توفير محتوى HTMLمخصص خاص بك من خلال الخاصية DocumentTextأو الخاصيات ذات الصلة.

الأحداث و التفاويض Events and Delegates

كل نموذج وأداة في الدوت نت تحتوي على إجراء ويندو WndProcالخاص بها،وكما أنها تعالج الرسالة الواردة من مضخة الرسائل،وتترجم هذه الرسائل إلى أحداث events. إن الأحداث *Events* هي تقنية دوت نت القياسية التي تتحكم-وجميع الفئات الأخرى-باستخدام كل من الفورم والأدوات.فعندما تُضمِّن فورم أو أداة في تطبيقك،تستطيع عرض واحد،أو بعض ،أو كل هذه الأحداث،وكتابة الكود الذي يستجيب بشكل مناسب.كل الكود المخصص الذي تكتبه لكل حدث يظهر في معالج الحدث*event handler*. ولكن ما الذي يحدث فعلياً بين إصبع المستخدم على الماوس والمنطق في معالج حدث مخصص؟.يبين الشكل التالي بشكل بياني ما يحث عملياً بين الإجراء(الفعل) والمنطق المخصص.



من الواضح ،أنه مايزال بعض الغموض mysteryمحيط بمعالج الحدث.الأدوات Controls –وكل الفئاتclasses - تحدد أي الأحداث ستجعل منها متاحة.فمن أجل الأدوات يتم إتاحة العديد من الأحداث بالتوازي مع إجراءات المستخدم الأولية:النقرClick، الماوس للأسفل MouseDown،ضغط مفتاح KeyPress، والحجم تغيرSizeChanged. ولكن يوجد أيضاً العديد من الأحداث يمكن أن يتم إطلاقها فقط من خلال التعديلات على الأداة خلال الكود المصدري: TabIndexChanged (عندما يتم تغير ترتيب مفتاح التنقل tab-key للأدوات)، BackgroundImageChanged و CursorChangedهي فقط ثلاث من العديد من الأحداث التي لايمكن للمستخدم أن يؤثر عليها بشكل مباشر.العديد من الأحداث الأخيرة ترتبط بالتغيرات على مستوى النظام system-level changes،مثل الحدث SystemColorsChanged،والذي يتم إطلاقه عندما يعدل المستخدم جدولة الألوان الواسعة للنظام system-wide color scheme من خلال لوحة التحكمcontrol panel .

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click

    End Sub
```

يستقبل معالج الحدث هذا معاملين نسبيين argumentsمن حدث تم إطلاقه:System.Object حالة(المرسلsender) وSystem.EventArgs نسخة (حالة e)، معالجات أحداث أخرى يمكن أن تستخدم معاملات نسبية مختلفة قليلاً،لذلك كيف تعرف ما تستخدم؟إن أي حدث معرف ضمن فئة أداة يجب أن يشير أيضاً إلى عدد ونوع المعاملات النسبية التي يرسلها إلى معالج الحدث.تتضمن الفيجوال بيسك العبارة Eventالتي تعرف الأحداث.مع أنه من الأرجح أنه تم كتابة الزر في السي شارب#C،إليك ما يبدو عليه تعريف حدث لحدث نقر الزر في الفيجوال بيسك:

```
Public Event Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
```

هذا التعريف يبدو مشابه كثيراً لمعالج الحدث السابق(Private Sub)،وهو كذلك.تؤسس العبارة Event عقد(اتفاقية)تمرير وسيط parameter- passing contract بين الأداة والكود الذي يريد أن يستقبل إشعارات(بيانات) الحدث event notifications.في هذه الحالة،حدث النقر يعد promisesأن يرسل معاملين نسبيين إلى معالج الحدث.الأول هو المرسلsender،وهو مرجع(دليل reference)للكائن الذي يشير إليه الحدث.من أجل أدوات الزرButton controls ، هذا الوسيط يستقبل مؤشر لنسخة(حالة) الزر نفسه.المعامل النسبي الثاني، e،يوفر طريقة لتمرير كائن كامل للمعلومات الإضافية.إن الفئة System.EventArgs ليس لديها الكثير من المعلومات،ولكن بعض الأحداث تستخدم تنوع ما في المعامل النسبي الثاني الذي يستخدم System.EventArgs كفئة قاعدية له. هذا يكشف أن المعاملات النسبية المستخدمة لحدث النقر هي مشتركة إلى حدما بين الأدوات والأحداث المختلفة.فعوضاً عن إعادة كتابة قائمة المعاملات النسبية في كل عبارة حدث،يمكن لمصمم الأداة أن يعرف تفويض له*delegate*: نوع ما للدوت نت يعمل على تعريف قائمة معاملات نسبية.أما من أجل الدالات ،قيمة راجعة.

```
Public Delegate Sub StandardEventDelegate(ByVal sender As System.Object, ByVal e As System.EventArgs)
```

يمكن لعبارات الحدث فيما بعد استخدام تفويض معرف كاختصار لقائمة الوسيطات الكاملة المكتوبة.

```
Public Event Click As StandardEventDelegate
```

سواء كان الحدث يستخدم تفويض أو قائمة المعاملات النسبية الكاملة،فله سيطرة تامة firm grasp على ما يحتاج أن يرسله من بيانات إلى أي معالج حدث مستقبل .وإنه يرسل هذه المعاملات النسبية باستخدام عبارة الفيجوال بيسكRaiseEvent. دعنا نقتفي أثر هذه المعالجة من أجل أداة الزر.عندما ينقر المستخدم على الزر، تعمل مضخة الرسائل message pump على إيجاد طريقة لترسل رسالة إلى إجراء WndProcالخاص بأداة الزر.وإن ذلك الزر يتفحص الرسالة،ويرى أنها نقر بالماوس ويقرر أن يخبر معالج الحدث event handlers حولها.ومن ثم،من ضمن كودWndProc، يعمل على إطلاق الحدث.

```
RaiseEvent Click(Me, New System.EventArgs)
```

تشير كلمة الفيجوال بيسك المحجوزة Meإلى حالة(نسخة)أداة الزر نفسها.لايحتوي المعامل النسبي e لأداة الزر أبعد من الحقول الافتراضية المضمنة في نسخة(حالة) System.EventArgs،لذلك فإن WndProcفقط يرسل حالة(نسخة) فارغة جديدة.الأدوات ذات حدث مع معاملات نسبية أخرى سيكون قد أنشئ حالة أولاً،يملأها بالبيانات المتعلقة،ويمرر تلك النسخة إلى معالج الحدث.إذا ما أطلق حدث في تطبيق،ولم يكن هناك معالج حدث يسمعه،هل يعمل صوت؟من المحتمل لا.لايوجد متطلبات لأن يكون لحدث ما معالجات فعالة مستقبلة(منصتة).ولكن عندما نريد أن نهتم بحدث،كيف نفعل هذا؟الطريقة القياسية لعمل هذا في تطبيق نماذج ويندوز هناك خطوتي معالجة.الأولى،يحتاج مستخدم الأداة(فئة فورمكyour form class) أن يعلن عن أداة،بمعنى"أريد أن أعرض أحداثك(أي أن الفورم ستعرض أحداث الأداة الموضوعة على سطحها)"ومن ثم توصل معالجات الأحداث بأحداث خاصة.سابقاً في هذا الفصل،رأينا أن إضافة أداة إلى واجهة المستخدم يحث عملياً على كتابة كود مصدر source code في الملف*Form1.designer.vb* . إليك الكود الذي يتم إضافته من أجل أداة زر مسماة Button1(مع أرقام الأسطر).

```
01 Partial Class Form1
02 Inherits System.Windows.Forms.Form
03
04 Friend WithEvents Button1 As System.Windows.Forms.Button
05
06 Private Sub InitializeComponent( )
07 Me.Button1 = New System.Windows.Forms.Button
08
09 Me.Button1.Location = New System.Drawing.Point(48, 16)
10 Me.Button1.Name = "Button1"
11 Me.Button1.Size = New System.Drawing.Size(75, 23)
12 Me.Button1.TabIndex = 0
13 Me.Button1.Text = "Button1"
14 Me.Button1.UseVisualStyleBackColor = True
15
16 Me.Controls.Add(Me.Button1)
17 End Sub
18 End Class
```

الكود في الطريقة InitializeComponentيعمل على إنشاء حالة لأداة الزر(السطر7)،يعمل على تعديل خاصياته للحصول على المظهر الذي نريد(السطر9 إلى السطر14)،يلحقه بالفورم من خلال السطر16.ولكن يوجد سطر واحد إضافي يعمل على تعريف حقيقة الزر كنوع لمتغير مرجعي reference type variable (السطر4) :

```
Friend WithEvents Button1 As System.Windows.Forms.Button
```

سنتحدث فيما بعد عن الحقول على مستوى الفئة class-level fields،والزر Button1هو حقل نموذجي على مستوى الفئة.ولكن الكلمة المحجوزة WithEventsالمضمنة في العبارة هي ما تتيح للأداة معرفة أن أحدما يريد عرض إشعارات الحدث.الآن ،في أي وقت ينطلق حدث الزر،فإنه يعرف أنه يمكن للفورم Form1أن تحتوي على معالجات حدث تراقب وتهتم(تستقبلwatching and listening).

الجزء الثاني للخطوتين:تتضمن عملية الوصل بين الحدث والمعالج event-to-handler connection process)الربط الحقيقي للمعالج).لننظر مرةً أخرى على تعريف معالج حدث النقر Click event handler لنسخة الزرButton1 .

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click

    End Sub
```

فهي مجرد طريقة فئة عادية،ولكن مع الشرط Handlesالمعلق في نهاية التعريف.هذا التعبير هو مايربط معالج الحدث event handler مع حدث Button1.Click نفسه.تستطيع إتباع الكلمة المحجوزة Handlesبعدة أسماء أحداث.

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click, Button2.Click, Button3.Click

    End Sub
```

الآن،سيستمع معالج حدث مفرد لنقر عدة أزرار من عدة أدوات مختلفة.بإمكانك حتى مزج الأحداث المعروضة،فيمكن لمعالج حدث واحد أن يستمع لأحداث مسماة مختلفة.

```
Private Sub ManyEvents(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.MouseDown,Button2.MouseUp
End Sub
```

تنوع آخر هو أن يكون لمعالجات حدث متعددة عرض حدث مفرد،على الرغم من أن على الفيجوال بيسك أن تقرر استدعاء أي معالج أولاً

```
  Private Sub FirstHandler(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
    End Sub

     Private Sub SecondHandler(ByVal sender As System.Object,ByVal e As System.EventArgs) Handles
Button1.Click
    End Sub
```

يوجد طريقة أخرى لوصل أحداث events بمعالجات أحداث event handlers والتي لاتتضمن سواء الكلمة المحجوزة WithEvents أو الكلمة المحجوزة Handles، وهذا ما يتيح لك السيطرة على ترتيب المعالجة لعدة معالجات حدث.حالما يكون لديك حالة لفئة تعرض حدث event-exposing class،فإنك تربط معالج إلى واحد من أحداثه باستخدام العبارة AddHandler. العبارة التالية تربط حدث نقر الزر Button1 بمعالج حدث مسمى MyHandler. يجب أن يكون للطريقة MyHandler قائمة معاملات نسبية محققة من أجل تعريف الحدث.

```
AddHandler Button1.Click, AddressOf MyHandler
```

تزيل العبارة المخصصة RemoveHandler اتصال معالج حدث بحدث ما.
سنمضي الكثير من الوقت في كتابة معالجات الأحداث.ولكنني سأدخل أيضاً ضمن جميع التفاصيل بحيث تستطيع أخذ فوائد جمة عن هذه التقنية.فلا تسمح لك الفيجوال بيسك من عرض الأحداث الخاصة بالأدوات فقط ولكن أيضاً تتيح لك تصميم أحداث جديدة في فئاتك الخاصة.بإمكانك استخدام الكلمات المحجوزة Delegate أو Event، RaiseEvent، WithEvents، Handles، AddHandler، RemoveHandler، من أجل أحداثك المخصصة،والتي يتم إطلاقها بواسطة شروط ما تختارها أنت.فإذا كان لديك فئة تمثل موظف employee،بإمكانك جعلها تطلق حدث"الطرد " Fired عندما يفقد الموظف عمله.بإضافة أحداث خاصة، فإنك تجعل من الممكن لكود خاص من أن يتم وصله بمنطق فئتك class logic،أو حتى ولو لم يكن لمبرمج ما صلاحية الدخول للكود المصدري التابع لفئتك.

جعل النماذج مفيدة Making Forms Useful

إن أي نموذج تعمل على تعريفه ككائن بدء لمشروعك يظهر بشكل آلي عندما يبدأ البرنامج.أما بقية النماذج الأخرى فتحتاج لعرضها بشكل يدوي،باستخدام إما الطريقة "إظهار Show" أو الطريقة"إظهار حوار ShowDialog " لتلك الفورم.مثلاً،إذا كان لديك فورم تدعى Form2 تستطيع عرضها باستخدام الطريقة Show :

```
Form2.Show()
```

تعرض الطريقة Show نموذج غير نمطي *modeless form*. النماذج غير النمطية يمكن الوصول إليها بشكل مستقل عن جميع الفورمات الأخرى في التطبيق المشغل.ويمكن لكل الفورمات الغير نمطية أن يتم تفعيلها في أي وقت بالنقر عليها فقط،والفورم التي تنقر عليها ستأتي إلى الواجهة أمام جميع الفورمات الأخرى وستستقبل تركيز الإدخال input focus. يمكن لبرنامج أن يكون له واحد،أو اثنان أو حتى الكثير من النماذج الغير نمطية تفتح في نفس الوقت،ويمكن للمستخدم أن يتنقل بينها بكل حرية.
النماذج النمطية *Modal forms* تتحكم بجميع الإدخالات في التطبيق،طالما أنها تظهر على الشاشة.والنماذج النمطية تدعى عادة"بالحوارات dialogs ".فعلى المستخدم أن يكمل النموذج النمطي ويغلقه قبل أن يتمكن من الوصول لفتح أي نموذج آخر في التطبيق.فنافذة صندوق الرسالة التي تظهر عندما تستخدم الدالة MsgBox هي نافذة حوار نمطية. تعرض الطريقة ShowDialog نماذج نمطية،وتتيح لك العودة بقيمة من تلك الفورم.والقيمة المعادة هي أعضاء العداد System.Windows.Forms.DialogResult.
لعرض نموذج نمطي استخدم طريقة الفورم ShowDialog وبشكل اختياري التقط قيمتها المعادة.

```
Dim theResult As DialogResult
  theResult = Form2.ShowDialog()
```

الحوارات النمطية Modal dialogs مفيدة لتحديث سجل ما يتطلب نقر على زر موافق OK عند اكتمال التغييرات .لنقول أنك كنت تكتب تطبيق يعرض قائمة كتب ألكسندر دامس.من المحتمل أن يتضمن نموذجين☹1)النموذج الرئيسي الذي يعرض قائمة الكتب،(2)النموذج الثانوي(الابن)والذي يتيح لك كتابة اسم كتاب مفرد. ألن يكون من العظيم إذا استطعت العودة باسم الكتاب(أو ربما،بالرقم المعرف ID لسجل ما خاص بالكتاب كما تم تخزينه في قاعدة البيانات)بدلاً من قيمة نتيجة الحوار DialogResult .
إذا كانت الطريقة ShowDialog، طريقة عامة public method لفئة فورم مضمنة،تستطيع العودة بنتيجة كود،من المحتمل أننا نستطيع إضافة طرق عامة أخرى للنموذج تعود بنتيجة كود بحيث يكون لديها معنى حقيقي.في الواقع،نستطيع ذلك.خذ الفورم الابن(المسمى BookEntry)مع حقل البيانات المدخلة(عنوان الكتاب BookTitle)،والأزرار OK (ActOK) و Cancel (ActCancel)،كما هو مبين في الشكل التالي.



عندما ندعمه بالكود التالي،هذا الفورم البسيط يعود بما تم كتابته في الحقل عندما ينقر المستخدم على زر موافق(في البداية يرفض القيم الفارغة)،أو يعود بنص فارغ عند النقر على إلغاء Cancel .

```
Public Class BookEntry
    Public Function EditTitle() As String
        ' ----- Show the form, and return what the user enters.
        If (Me.ShowDialog() = DialogResult.OK) Then
            Return BookTitle.Text.Trim
        Else
            Return ""
        End If
    End Function
    Private Sub ActCancel_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ActCancel.Click
        ' ----- Return a blank title for "Cancel."
        Me.DialogResult = DialogResult.Cancel
        ' ----- Continue with EditTitle( )
```

```
    End Sub
    Private Sub ActOK_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ActOK.Click
        ' ----- Only accept valid titles.
        If (Len(BookTitle.Text.Trim) = 0) Then
            MsgBox("Please supply a valid title.")
        Else
            Me.DialogResult = DialogResult.OK
            ' ----- Continue with EditTitle( )
        End If
    End Sub
End Class
```

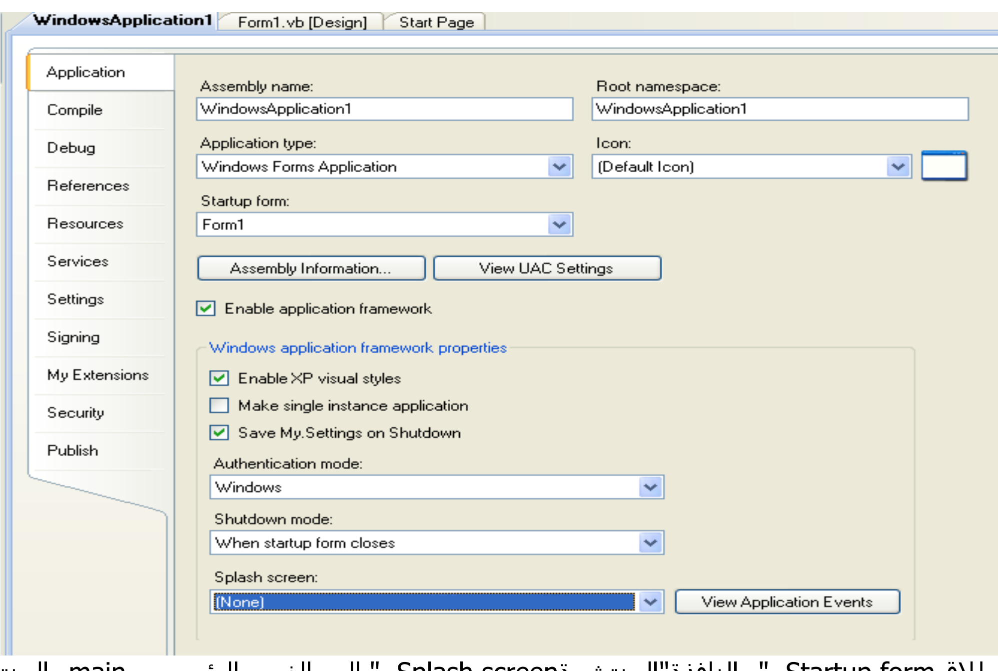
لاستخدام هذه الفورم،تستدعي الفورم الرئيسي الطريقةEditTitle، والتي تعود بعنوان الكتاب المدخل من قبل المستخدم.

```
Dim newTitle As String = BookEntry.EditTitle()
```

يعرض الروتين EditTitleالفورم بشكل نمطي form modally من خلال الطريقة ShowDialog،وينتظر هناك فقط حتى يغلق المستخدم الفورم.إغلاق الفورم يتم من خلال الزر موافق أو الزر إلغاء،وإعداد خاصية DialogResult للفورم لديه تأثير جانبي على إغلاق الفورم. حالما يتم إغلاق الفورم،يتم إرجاع تنفيذ إلى EditTitle،والذي يعمل تفحص سريع قبل العودة بالقيمة النهائية.وبالتالي نحصل عليها:واجهة عامة جديدة من أجل قيمة النماذج المعادة الأكثر أهمية.سنستخدم هذه الطريقة كثيراً في تطبيق مشروع المكتبة.

مشروع Project

كود مشروع هذا الفصل ينفذ الفورم الرئيسي الأساسي Main لمشروع المكتبة،بالإضافة إلى الفورم "Splash "التي تظهر عندما يشتغل المشروع للمرة الأولى.تعرف ميكروسوفت أن هذه كانت حاجة مشتركة،متضمنة دعم لكل من الفورمات الرئيسية والمنتشرة في نظام إطار عمل تطبيق الفيجوال بيسك. بشكل افتراضي،هذا النظام يتم تمكينه من خلال صفحات خاصيات المشروع(شاهد الشكل التالي).



تشير حقول"فورم الانطلاقStartup form " والنافذة"المنتشرةSplash screen " إلى الفورم الرئيسي main والمنتشرsplash على الترتيب.فهذا سريع،وسهل،وهو مجرد شيء معد للاستخدام،لذلك لندخل في العمل.
سيكون الآن وقت عظيم لتحميل المشروع البادئ للفصل السابع.

تركيب الشاشة المنتشرة Configuring the Splash Screen

لقد عملت على إضافة ملف فورم جديد للمشروع،مسمى *Splash.vb* (أي أن الفورم نفسه مسمى Splash)،متضمناً بعض العناصر المعروضة.تحرى عن الصورة على الفورم.يتم جلبها من خلال أداة"صندوق الصورة PictureBox "،ولكن تم تخزينها في التطبيق كمصدر *resource*،يتم إلحاق مجموعة من السلاسل الحرفية stringsوالصور imagesلكود المصدر.يتضمن مجلد المصادر *Resources* فيمستكشف الحلول ملف هذه الصور.ويتم وصله ضمن صندوق الصورة من خلال خاصية Imageللأداة. ومن المؤكد أنها تجعل الفورم ذات مظهر جيد.عملك يكون ربط هذه الفورم ضمن تتابع بدء startupالتطبيق.
اذهب إلى خاصيات المشروع(المعروض في الشكل في الأعلى)،ضع حقل " Splash screen "إلى Splash.وهذا له تأثير جانبي على إعداد My.Application.SplashScreen لفورم Splash.والآن شغل البرنامج.سترى أن النافذة المنتشرة تظهر حوالي 1/100من الثواني،وبسرعة يتم استبدالها بواسطة الفورم الرئيسي.
كما رأيت تعديل حقل" Splash screen " يؤدي إلى ظهور النافذة المنتشرة splash screen بإيجاز.ولكن سيحفظها التطبيق ظاهرة حتى يكتمل عقد التحضير الكافي للفورم الرئيسي.بما أننا لم نعمل أي تحضير،فإن التطبيق يعرض الفورم الرئيسي حالاً.

أخيراً،سنضيف حزمة كود بدء خاص بقاعدة البيانات bunch of database-related startup code والذي سيأخذ وقت أكثر.ولكن حالياً،سنعمل على تزيفهfake .في وحدة model إطار العمل للتطبيق،أي كود تريد معالجته عندما يبدأ البرنامج الظهور للمرة الأولى يتم في حدث البدء(Startup)
وهذا الحدث هو واحد من التجميعات الصغيرة لأحداث مضمنة مع التسلسل الهرميMy. يظهر الكود المصدري لهذه الأحداث في ملف *ApplicationEvents.vb* file،الملف الذي تضيفه الفيجوال بيسك بشكل آلي لمشروعك عند الحاجة.استخدم زر "عرض أحداث التطبيق View Application Events " في صفحات خصائص المشروع project properties ومن تبويب تطبيق Applicationلفتح ملف الكود المصدري.

```
Namespace My
    Partial Friend Class MyApplication

    End Class

End Namespace
```

لنفرض أن المتطلبات الأولية لمشروع المكتبة تأخذ حوالي ثلاث ثواني.تتضمن الدوت نت الطريقةSleep والتي تؤجل الكود لعدد معين من الملي ثانية.

```
Namespace My
    Partial Friend Class MyApplication

        Private Sub MyApplication_Startup(ByVal sender As Object, ByVal e As
Microsoft.VisualBasic.ApplicationServices.StartupEventArgs) Handles Me.Startup
            ' ----- Take a three-second nap.
            System.Threading.Thread.Sleep(3000)
        End Sub
    End Class

End Namespace
```

شغل البرنامج الآن،وسترى أن النافذة المنتشرة تبقى حوالي ثلاث ثواني(3000ملي ثانية).بالتالي توجد فرصة جيدة لأن يأخذ كل من قاعدة البيانات وكود بدء التطبيق وقت أقل من ثلاث ثواني،أعني،نحن نتكلم عن خادم سكول SQL هنا،من المفترض أن يتوهج بسرعة blazing fast.
لذلك من الواضح أننا مانزال بحاجة لتأجيل انتقال النافذة المنتشرة.إن الكائن My.Application يحدث لأن يتضمن الخاصية الأفضل التي نحتاجها لنؤكد على التأجيل.تشير الخاصية MinimumSplashScreenDisplayTime إلى العدد الأقل من الملي ثانية الذي يجب أن يتم عرض الشاشة المنتشرة.القسم السيئ هو أن عليك إسنادها في مكان غريب حقاً،على الأقل عند المقارنة إلى كيفية برمجة الفيجوال بيسك التي تعلمناها حتى الآن.
احذف كل الكود الذي عملنا على إضافته حتى الآن،أي احذف الطريقة MyApplication_Startup بالكامل.ومن ثم ضع الكود التالي مكانه.

```
Namespace My
    Partial Friend Class MyApplication
        Protected Overrides Function OnInitialize (ByVal commandLineArgs As System.Collections. _
ObjectModel.ReadOnlyCollection(Of String)) As Boolean
            ' ----- Display the splash form for at least 3 seconds.
            My.Application.MinimumSplashScreenDisplayTime = 3000
            Return MyBase.OnInitialize(commandLineArgs)
        End Function
    End Class
End Namespace
```

يحتوي مقطع الكود على الكثير من الأشياء التي لم نتكلم عنها حتى الآن،ولن أتكلم عنها حتى عدة فصول لاحقة.يكفي القول أن الطريقة OnInitialize هي واحدة من أول الأشياء التي تحدث في عمر البرنامج،وأن هذا هو المكان الذي يتم فيه إسناد MinimumSplashScreenDisplayTime.
الشيء الأخير الذي عليك عمله للشاشة المنتشرة splash screen هو تضمين بعض الكود الذي يعرض رقم النسخة version number.لقد عملنا هذا سابقاً من أجل نموذج في الفصل السابق،لذلك سنعمل هنا على إضافة كود مشابه لمعالج حدث تحميل Load event فورمSplash. وسنحدث رسالة حقوق الطباعةcopyright، أيضاً،افتح الكود المصدري لفورم Splashوأضف الكود التالي:

```
Public Class Splash
    Private Sub Splash_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
        ' تحديث رقم الاصدار
        With My.Application.Info.Version
            ProgramVersion.Text = "Version " & .Major & "." & _
            .Minor & " Revision " & .Revision
        End With
        ProgramCopyright.Text = My.Application.Info.Copyright
    End Sub
End Class
```

الآن شغل البرنامج ولاحظ كيف تنتظر فترة محددة ومن ثم تظهر الفورم الرئيسية.

تركيب الفورم الرئيسية Configuring the Main Form

على الرغم من أننا صممنا الفورم الرئيسي في فصل سابق،فهو متناثر نسبياً،وتتضمن فقط زر واحد هو"حول About ".مشروع هذا الفصل يضيف كل عناصر واجهة المستخدم للفورم.في الحقيقة،لقد عملت على إضافة الأدوات لسطح تلك الفورم من أجلك(شاهد الشكل التالي).ولكن تستطيع إضافة بعض معالجات الحدث التي تجعل الفورم تعرض بعض حيويتهاpizzazz .



كل كود الحدث العام من أجل الفورم يظهر كما يلي.

```
Private Sub ActLibraryItem_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles ActLibraryItem.LinkClicked
        ' ----- Library Item mode.
        TaskLibraryItem()
    End Sub
    Private Sub PicLibraryItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles PicLibraryItem.Click
        ' ----- Library Item mode.
        TaskLibraryItem()
    End Sub
    Private Sub ActPatronRecord_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles ActPatronRecord.LinkClicked
        ' ----- Patron Record mode.
        TaskPatronRecord()
    End Sub
    Private Sub PicPatronRecord_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles PicPatronRecord.Click
        ' ----- Patron Record mode.
        TaskPatronRecord()
    End Sub
    Private Sub ActHelp_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles ActHelp.LinkClicked
        ' ----- Help mode.
        TaskHelp()
    End Sub
    Private Sub PicHelp_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
PicHelp.Click
        ' ----- Help mode.
        TaskHelp()
    End Sub
    Private Sub ActCheckOut_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles ActCheckOut.LinkClicked
        ' ----- Check Out mode.
        TaskCheckOut()
    End Sub
    Private Sub PicCheckOut_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles PicCheckOut.Click
        ' ----- Check Out mode.
        TaskCheckOut()
    End Sub
    Private Sub ActCheckIn_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles ActCheckIn.LinkClicked
```

```
        ' ----- Check In mode.
        TaskCheckIn()
    End Sub
    Private Sub PicCheckIn_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles PicCheckIn.Click
        ' ----- Check In mode.
        TaskCheckIn()
    End Sub
    Private Sub ActAdmin_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles ActAdmin.LinkClicked
        ' ----- Administration mode.
        TaskAdmin()
    End Sub
    Private Sub PicAdmin_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
PicAdmin.Click
        ' ----- Administration mode.
        TaskAdmin()
    End Sub
    Private Sub ActProcess_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles ActProcess.LinkClicked
        ' ----- Daily Processing mode.
        TaskProcess()
    End Sub
    Private Sub PicProcess_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles PicProcess.Click
        ' ----- Daily Processing mode.
        TaskProcess()
    End Sub
    Private Sub ActReports_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles ActReports.LinkClicked
        ' ----- Print Reports mode.
        TaskReports()
    End Sub
    Private Sub PicReports_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles PicReports.Click
        ' ----- Print Reports mode.
        TaskReports()
    End Sub
    Private Sub TaskLibraryItem()
        ' ----- Library item task.

        ' ----- Update the display.
        AllPanelsInvisible()
        PanelLibraryItem.Visible = True
        ActLibraryItem.BackColor = SystemColors.Control
        LabelSelected.Location = New System.Drawing.Point(LabelSelected.Left, PicLibraryItem.Top)
        Me.AcceptButton = ActSearch

        ' ----- Get ready for a new search.
        On Error Resume Next
        SearchText.Focus()
    End Sub
    Private Sub TaskPatronRecord()
        ' ----- Patron Record mode.

        ' ----- Update the display.
        AllPanelsInvisible()
        PanelPatronRecord.Visible = True
        ActPatronRecord.BackColor = SystemColors.Control
        LabelSelected.Location = New System.Drawing.Point(LabelSelected.Left, PicPatronRecord.Top)
        Me.AcceptButton = ActAccessPatron
    End Sub
    Private Sub TaskHelp()
        ' ----- Help mode.

        ' ----- Update the display.
```

```
        AllPanelsInvisible()
        PanelHelp.Visible = True
        ActHelp.BackColor = SystemColors.Control
        LabelSelected.Location = New System.Drawing.Point(LabelSelected.Left, PicHelp.Top)
        Me.AcceptButton = Nothing
    End Sub
    Private Sub TaskCheckOut()
        ' ----- Check Out mode.

        ' ----- Update the display.
        AllPanelsInvisible()
        PanelCheckOut.Visible = True
        ActCheckOut.BackColor = SystemColors.Control
        LabelSelected.Location = New System.Drawing.Point(LabelSelected.Left, PicCheckOut.Top)
        Me.AcceptButton = ActDoCheckOut
    End Sub
    Private Sub TaskCheckIn()
        ' ----- Check In mode.

        ' ----- Update the display.
        AllPanelsInvisible()
        PanelCheckIn.Visible = True
        ActCheckIn.BackColor = SystemColors.Control
        LabelSelected.Location = New System.Drawing.Point(LabelSelected.Left, PicCheckIn.Top)
        Me.AcceptButton = ActDoCheckIn

        ' ----- Reset the input fields.
        CheckInDate.Value = Today
        CheckInDate.MaxDate = Today
        CheckInDate.MinDate = DateAdd(DateInterval.Day, -6, Today)
        CheckInDay.Text = "Today"
        CheckInDay.BackColor = SystemColors.Control
        CheckInDay.ForeColor = SystemColors.ControlText
        CheckInBarcode.Text = ""
        LabelCheckedIn.Visible = False
        CheckedInDetail.Visible = False
        CheckedInDetail.Text = "No information available."
        CheckInBarcode.Focus()
    End Sub
    Private Sub TaskAdmin()
        ' ----- Administration mode.

        ' ----- Update the display.
        AllPanelsInvisible()
        PanelAdmin.Visible = True
        ActAdmin.BackColor = SystemColors.Control
        LabelSelected.Location = New System.Drawing.Point(LabelSelected.Left, PicAdmin.Top)
        Me.AcceptButton = Nothing
    End Sub
    Private Sub TaskProcess()
        ' ----- Daily Processing mode.

        ' ----- Update the display.
        AllPanelsInvisible()
        PanelProcess.Visible = True
        ActProcess.BackColor = SystemColors.Control
        LabelSelected.Location = New System.Drawing.Point(LabelSelected.Left, PicProcess.Top)
        Me.AcceptButton = ActDoProcess
    End Sub
    Private Sub TaskReports()
        ' ----- Print Reports mode.

        ' ----- Update the display.
        AllPanelsInvisible()
        PanelReports.Visible = True
        ActReports.BackColor = SystemColors.Control
```

```
        LabelSelected.Location = New System.Drawing.Point(LabelSelected.Left, PicReports.Top)
        Me.AcceptButton = ActDoReports
    End Sub
    Private Sub MainForm_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles
MyBase.Load
        ' ----- Prepare the form.

        ' ----- Adjust the various panel positions.
        PanelLibraryItem.Location = New Point(TaskListBackground.Width, WelcomeBackground.Height)
        PanelPatronRecord.Location = PanelLibraryItem.Location
        PanelHelp.Location = PanelLibraryItem.Location
        PanelCheckOut.Location = PanelLibraryItem.Location
        PanelCheckIn.Location = PanelLibraryItem.Location
        PanelAdmin.Location = PanelLibraryItem.Location
        PanelProcess.Location = PanelLibraryItem.Location
        PanelReports.Location = PanelLibraryItem.Location
        PanelLibraryItem.Visible = True
        ActSearchLimits.PerformClick()
    End Sub
    Private Sub AllPanelsInvisible()
        ' ----- Hide all work panels.
        PanelLibraryItem.Visible = False
        PanelPatronRecord.Visible = False
        PanelHelp.Visible = False
        PanelCheckOut.Visible = False
        PanelCheckIn.Visible = False
        PanelAdmin.Visible = False
        PanelProcess.Visible = False
        PanelReports.Visible = False

        ' ----- Correctly color all action labels.
        ActLibraryItem.BackColor = Color.White
        ActPatronRecord.BackColor = Color.White
        ActHelp.BackColor = Color.White
        ActCheckOut.BackColor = Color.White
        ActCheckIn.BackColor = Color.White
        ActAdmin.BackColor = Color.White
        ActProcess.BackColor = Color.White
        ActReports.BackColor = Color.White
    End Sub
    Private Sub SearchText_Enter(ByVal sender As Object, ByVal e As System.EventArgs) Handles
SearchText.Enter
        ' ----- Highlight the entire text.
        SearchText.SelectAll()
    End Sub
    Private Sub ActSearchLimits_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles ActSearchLimits.Click
        ' ----- Toggle the display of the media type and location fields.
        If (ActSearchLimits.Top = LabelMoreLimitsTop.Top) Then
            ' ----- Hide the two fields.
            LabelSearchMediaType.Visible = False
            SearchMediaType.Visible = False
            LabelSearchLocation.Visible = False
            SearchLocation.Visible = False
            ActSearchClear.Location = New System.Drawing.Point(ActSearchClear.Left,
SearchMediaType.Top)
            ActSearchLimits.Location = New System.Drawing.Point(ActSearchLimits.Left,
SearchMediaType.Top)
            ActSearch.Location = New System.Drawing.Point(ActSearch.Left, SearchMediaType.Top)
            ActSearchLimits.Text = "Mor&e >>"
        Else
            ' ----- Show the two fields.
            ActSearchClear.Location = New System.Drawing.Point(ActSearchClear.Left,
LabelMoreLimitsTop.Top)
            ActSearchLimits.Location = New System.Drawing.Point(ActSearchLimits.Left,
LabelMoreLimitsTop.Top)
```

```
            ActSearch.Location = New System.Drawing.Point(ActSearch.Left, LabelMoreLimitsTop.Top)
            LabelSearchMediaType.Visible = True
            SearchMediaType.Visible = True
            LabelSearchLocation.Visible = True
            SearchLocation.Visible = True
            ActSearchLimits.Text = "<< L&ess"
        End If
    End Sub
    Private Sub CheckInBarcode_Enter(ByVal sender As Object, ByVal e As System.EventArgs) Handles
CheckInBarcode.Enter
        ' ----- Highlight the entire text.
        CheckInBarcode.SelectAll()
    End Sub
    Private Sub CheckOutBarcode_Enter(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckOutBarcode.Enter
        ' ----- Highlight the entire text.
        CheckOutBarcode.SelectAll()
    End Sub
    Private Sub MainForm_KeyDown(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyEventArgs) Handles MyBase.KeyDown
        ' ----- The keys F2 through F9 access the different sections of the form.
        Select Case (e.KeyCode)
            Case Keys.F2
                TaskLibraryItem()
                e.Handled = True
            Case Keys.F3
                TaskPatronRecord()
                e.Handled = True
            Case Keys.F4
                ' ----- Allow form to handle Alt+F4.
                If (e.Alt = True) Then
                    Me.Close()
                Else
                    TaskHelp()
                End If
                e.Handled = True
            Case Keys.F5
                TaskCheckOut()
                e.Handled = True
            Case Keys.F6
                TaskCheckIn()
                e.Handled = True
            Case Keys.F7
                TaskAdmin()
                e.Handled = True
            Case Keys.F8
                TaskProcess()
                e.Handled = True
            Case Keys.F9
                TaskReports()
                e.Handled = True
        End Select
    End Sub
```

معظم هذا الكود يوجد لينقل أشياء تخص العرض.على سبيل المثال،يمكن للمستخدم أن يصل للميزات المختلفة للفورم بالنقر على الأيقونات أو لصاقات النصوص ذات الصلة على طول الجهة اليسارية من الفورم.كل أيقونة ولصاقة تطلق واحد من سبع روتينات(إجراءات) والتي توجد لترتيب الأساس.الأيقونة اليسارية الأكبرPicLibraryItem ، تستدعي الروتين المشترك TaskLibraryItemعندما يتم النقر عليها.

الإجراء TaskLibraryItemيضبط اللوحات المتنوعة والحقول على العرض بحيث يرى المستخدم هذه الحقول المطلوبة للبحث في بنود المكتبة.

الإجراء AllPanelsInvisibleيعمل أيضاً بعض التعديلات على الشاشة.

أحب تضمين النص الموجود في حقل صندوق نص TextBoxمختار عندما يصبح أداة فعالة.كل أداة نص textتتضمن الطريقة SelectAll والتي تحقق هذا العمل. سوف نستدعي تلك الطريقة من خلال كل حدث دخول Enterلأداة صندوق نصTextBox، الحدث الذي يحدث عندما تستقبل أداة تركيز إدخال لوحة المفاتيحkeyboard. راجع الكود السابق(Private Sub SearchText_Enter).

إن استخدام الماوس من أجل الوصول إلى الميزات المختلفة للفورم شيء جيد،ولكن للتعامل مع كيبورد(لوحة مفاتيح)المستخدم يجب أن نضيف الكود الذي يدعم ميزات الوصول باستخدام المفاتيح من F2إلى F9.(راجع الكود السابق (Private Sub MainForm_KeyDown)).

فكل ضرب مفتاح keystrokeيأتي إلى معالج حدثKeyDown، وإن عبارة Select Case تختبره.وعندما يطابق حالة مدخلة ،فإن الكود ضمن مقطع الحالة Caseيتم تنفيذه.ضغط المفتاح F2يحرر الكود في مقطع الحالةKeys.F2 .والمفاتيح هي واحدة من العديد من العدادات الجاهزة والتي تستطيع استخدامها في تطبيقات الدوت نت.

لاحظ الكود الخاص بالنسبة للمفتاح F4.فهو يسمح للضم Alt-F4من الخروج من التطبيق،والذي هو ضم مفتاحي قياسي بالنسبة لبرامج الويندوز الموجودة.

عادة،جميع ضربات المفاتيح keystrokesتذهب إلى الأداة الفعالة،وليس للفورم.لتمكين معالج الحدث MainForm.KeyDown،يجب أن يتم وضع خاصية الفورم KeyPreviewإلى صحTrue .ضع هذه الخاصية وذلك بالرجوع إلى مصمم الفورم وصندوق الخصائص.

جعل البرنامج نسخة مفردة(حالة مفردة)Making the Program Single-Instance.

مشروع المكتبة يتم تصميمه للاستخدام فقط ضمن مكتبة صغيرة،فهو سيعمل فقط على شبكات عمل workstationsقليلة في نفس الوقت،ويمكن أن يصل حتى 10على الأكثر.وليس هناك حاجة لتشغيل أكثر من نسخة واحدة على شبكة مفردة،بما أن كل نسخة تتضمن جميع ميزات التطبيق الممكنة.واحدة من الميزات الجامدة المضمنة مع الفيجوال بيسك هي إمكانية إنشاء"تطبيق ذو حالة مفردةsingle-instance application "،والتي تجبر على إدارة تشغيل واحد في نفس الوقت على كل محطة (شبكة عمل).على الرغم من أنه كان بإمكانك إنشاء مثل هذه التطبيقات من قبل،ولكن الآن متاحة من خلال نقرة ماوس وحيدة.

لجعل مشروع المكتبة تطبيق وحيد الحالةsingle-instance application، اعرض صفحات خاصيات المشروعthe project properties، ومنه لوحة التطبيقApplication، ومن ثم اختر الحقل" اجعل التطبيق حالة مفردةMake single instance application ".عندما يحاول المستخدم تشغيل نسخة ثانية ،فإن الدوت نت سترفض تنفيذ المطلوب.بدلاً عن ذلك،ستطلق حدث التطبيقStartupNextInstance. وأية معالجة خاصة ترغب في إتمامها على التشغيل الثاني للحالة سيتم عملها في هذا المعالج.مثل معالج حدث الانطلاقStartup، يظهر معالج الحدث StartupNextInstanceفي الملف *ApplicationEvents.vb*.

من أجل مشروع المكتبة،الشيء الوحيد ا لحقيقي الذي نريد عمله حقاً عندما يحاول المستخدم تشغيل حالة ثانية هو التأكيد على أن التطبيق معروض من قبل وهو مركزي.بحيث يستطيع المستخدم عرضه بسهولة .افتح الملف*pplicationEvents.vb* وأضف معالج الحدث StartupNextInstance.

```
Private Sub MyApplication_StartupNextInstance(ByVal sender As Object, ByVal e As
Microsoft.VisualBasic.ApplicationServices.StartupNextInstanceEventArgs) Handles
Me.StartupNextInstance

            ' ----- Force the main form to the front.
            My.Application.MainForm.Activate()
        End Sub
```

Classes and Inheritance الفئات والوراثة

إن مبادئ ال ة ا وجهة بالكائنات((OOP) (*object-oriented programming*) هي حجر الأساس لنظام الدوت نت.وبدون (OOP) سيكون من الصعب دعم ا يزات ا وهرية للدوت نت،مثل الكائن ا ركزيSystem.Object، والذي هو حجر الأساس basic foundationلنظام الدوت نت.

على الرغم من أن قد ذكرت باختصار مبادئ التطوير بال ة ا وجهة بالكائنات الفصل الأول والثا ،فكانت فقط لتوف بعض السياق من أجل مناقشة مواضيع أخرى.ولكن هذا الفصل،لن أكتم أكثر من ذلك.فبعد ا ناقشة القوية بادئ ال ة ا وجهة بالكائنات،سأناقش كيف تستطيع استخدام هذه ا بادئ كود الدوت نت.

Object-Oriented Programming Conceptsمبادئ ال ة ا وجهة بالكائنات

إن ا عا CPU كمبيوترك يعا عبارات ا نطق والبيانات بالطريقة الشكلية القد ة:خطوة كل مرة،ينتقل خلال مناطق خاصة الذاكرة تكون موجهة بواسطة ا نطق،يعا القيم ا ستقلة والبتات تبعاً لنفس عبارات ا نطق هذه.فهو لايرى البيانات ككائنات معة،فهو يرى فقط البتات bits والبيتاتbytes .

إن الكود النهائي عندما يتم تنفيذه بواسطة معا ك CPUلن يكون بعد الآن موجه بالكائنات فهو كأي كود دوس MS-DOS قد .ولكن مترجم اللغة ا وجهة بالكائنات يوفر ذلك الوهم بأن ال ة كائنية التوجه مبنية ضمن الكمبيوتر.إنك تصمم كودك وبياناتك ط الكائنات،وا ترجم يأخذها من هناك. فهو(ا ترجم) يز كودك وبياناتك،يضيف بعض الكود الإضا لعمل مضاهاة(مقاربة) لسحر ال ة ا وجهة بالكائنات،و زمها ملف تنفيذي EXE file.تستطيع كتابة أي برنامج بال ة الكائنية التوجه باستخدام اللغات الإجرائية العاديةordinary procedural languages، أو لغة التجميع assembly language،ولكن التطبيقات ال تركز على البيانات كن أن يتم كتابتها بفعالية أك باستخدام فوائد تطوير اللغة الكائنية التوجهOOP .

The Object الكائن

جوهر ال ة كائنية التوجه object-oriented programming بالطبع هو الكائن*object*. فالكائن هو شخص ما،أو مكان أو أي شيء.ولكن هذا اسم noun،نعم إن الكائن يشبه الاسمnoun. فالكائنات هي تركيبات لبيانات ومنطق الكمبيوتر وال تشفر(ترمز) كينونات العا ا قيقي،مثل الناس، الأماكن،أو الأشياء،تستطيع أن لك كائنات ثل ناس،موظف ، منازل،...إ .

توفر الكائنات وسائل ا يات مناسبة لشرح وإدارة البيانات ا صاحبة لواحد من كائنات العا ا قيقي هذهreal-world objects. على سبيل ا ثال،إذا كان لديك موعة من الكائنات ثل الديفيديDVD، مثل عنوانه،ا مثل الذين مثلوا ا توى،طول الفيديو بالدقائق،فيما إذا الفيديو قد تضرر أو خدش، نه، وهكذا.إذا وصلت تطبيقك شغل الديفيدي على نظامك، كن لكائنك أن يتضمن ح على ا يزة "تشغيل"ال (تفترض أن قرص الديفيدي موضوع السواقة)ستبدأ بتشغيل الفيلم،من ا تمل من موقع بدء دد بالوقت أو من قسم الفيديو.

تعمل الكائنات بشكل جيد بسبب مقدرا ا على مقاربة(اكاة) ميزات نظ ا ا العا ا قيقي من خلال وسائل تطوير ال يات.وهي تقوم ذا من خلال أربع مواصفات رئيسية للكائن:التجريد*abstraction*، التغليف*encapsulation*، الوراثة*inheritance*، وتعدد الأشكال*polymorphism*.خلال هذا الفصل،يش ا صطلح "كائن *object*" عادةً إ حالة(نسخة)من شيء ما،استخدام خاص الذاكرة لعنصر معرف،حالة مع موعتها ا اصة من البيانات،وليس فقط تعريفه أو تصميمه.تش الفئة *Class* إ التصميم والكود ا صدري للكائن،متضمنة التنفيذ*implementation* .

Abstractionالتجريد

يدل التجريد على عرض دود لكائن من كائن العا ا قيقي.كطباعة ردة،يظهر الكائن ا ر الضروريات الأساسية كافئه من العا ا قيقي(شاهد الشكل التا).

كن للكائنات أن ثل بإتقان نظ ا من العا ا قيقي.بدلاً من ذلك،فهي تنفذ زين البيانات ومعا ة فقط على عناصر ا ثيل(النظ)من العا ا قيقي وال تكون هامة للتطبيق.ال يات ليست الشيء الوحيد ال تتطلب التجريد.فالتخطيط الط عند مكتب دكتورك هو ريد مل صحتك ا سدية.وعندما تشتري م ل جديد، فتقرير ا فتش هو ريد للحالة ا قيقية للبناء.وح قياس ا راري ردهة م لك هو ريد،فهو لاينقل بدقة كل التغ ات ا رارية الثانوية ا وجودة حول زجاجة الزئبق،عوضاً عنه، مع يع ا علومات ال كنه،وينقل نتيجة رقمية وحيدة.

كل هذه الأدوات ا ردة تسجل،أو تعمل وفق،أو تنقل فقط ا علومات الضرورية ال تصميمها لإدار ا.كائن ال يات software object

بنفس الطريقة، زن فقط،أويعمل وفق،أو ينقل ا علومات الضرورية حول نظ ه من العا ا قيقي.على سبيل ا ثال،إذا كنت تصمم كائن يراقب حالة بناء، من ا مكن أن تسجل التا :

1.موقع البناء وعنوانه.

2.مواد البناء الرئيسية(خشب،ا رسانة،العارضة ا عدنية،ا)

3.العمر(بالسنوات)

4.ا الة العامة(من قائمة اختيارات).

5.ملاحظات ا فتش.

على الرغم من أن البناء سيكون له لون،عدد الأبواب والنوافذ،والارتفاع،فهذه العناصر كن أن لاتكون هامة للتطبيق،وهكذا فلن تكون جزء من التجريد.هذه القيم ال تضمينها الكائن تدعى ا اصيات *properties*.إن أي قواعد معا ة processing rules أوحسابات calculations تواة ضمن الكائن الذي يعمل وفق ا اصيات properties (أو وفق بيانات خارجية أو داخلية مزودة)تعرف بالطرقmethods . مع بعضها ،الطرق وا اصيات تركب(تكون)أعضاء الكائنmembers of the object .

التغليف Encapsulation

الفائدة العظيمة من ال يات هي أنه كن للمستخدم أن يقوم بالكث من العمل ا عقد بسرعة وبسهولة.عملياً،تأخذ ال يات اهتمام بالسرعة والتعقيد من أجل ا ستخدم،و عدة حالات،لايهتم ا ستخدم بكيفية إ ام العمل.فترى أحدهم يقول"لا أعرف ولا أبا كيف تعمل الكمبيوترات طا ا أ ا تعطي النتائج ال احتاجها" وهي عبارة مشتركة تسمعها كث من الأحيان.وهي عبارة واقعية أيضاً، ا أن الكمبيوتر قد غلف *encapsulated* البيانات الضرورية ومنطق ا عا ة، لإ از ا همات ا رغوبة.

التغليف معزز بفكرة "الواجهات *interfaces*".على الرغم من أن الكمبيوتر كن أن توي على الكث من ا نطق ا فيد والبيانات،فإذا يكن هناك طريقة للتفاعل مع ا نطق أو البيانات،فإن الكمبيوتر سيكون أساساً عد النفع.توفر الواجهات الوسائل ال تتفاعل مع الكميات الداخلية لكائن ما.توفر الواجهة مدخلات و رجات سيطرة كمة ضمن البيانات والإجراءات الروتينية ا تواة ضمن كائن ما.وكمستهلك للكائن،فإنك غ معن بكيفية قيام الكائن بعمله داخلياً،طا ا أنه يعطي النتيجة ال تتوقعها من خلال واجهاته العامة ا عروضة.

استخدام الكمبيوتر كمثال،تتضمن الواجهات ا تنوعة (من ب الأشياء الأخرى)لوحة ا فاتيحkeyboard، أو الشاشةdisplay، ا اوسmouse، موصل الطاقةpower connector، USB، 1394منفذports، السبيكراتspeakers ، وصلة ا يكروفونmicrophone jack، وزر الطاقةpower button .

أغلب،الأشياء ال أوصلها مع هذه الواجهات هي أيضا صناديق سوداءblackboxes، وأخ اً التغليفات مع واجهات عامة معرفة بشكل كم.الطابعة غامضة بالنسبة .كيف كن شغل الطابعة printer driver أن يرسل أمر من خلال كبل اليواس USBوأخ اً تنشر ا ضمن ورقة قياس مع ،فهو غ قابل للتفسير inexplicable، ولكن لا أعرف ولا أبا كيف يتم هذا العمل فعلياً،طا ا أن الطابعة تنجز عملها.

الوراثة Inheritance

الوراثة الدوت نت لاتشبه الوراثة ا ياة العادية،ولكن كما ا ياة العادية دد الوراثة العلاقة ب كائن تلف .بشكل خاص،فهي دد كيف ينحدر كائن من كائن أخر.الفئة الأصلية original class علاقة الكائن تدعى الفئة القاعديةbase class .وهي تتضمن أعضاء واجهة متنوعة وغ دة،بالإضافة إ تفاصيل التنفيذ الداخلي.يتم تعريف الفئة ا شتقة derivedباستخدام الفئة القاعدية كنقطة بدء.الفئات ا شتقة ترث ا يزات من الفئة القاعدية.بشكل افتراضي،أي عضو عام معروض للفئة القاعدية يصبح بشكل آ عضو عام معروض للفئة ا شتقة،متضمناً التنفيذ. كن أن تار الفئة ا شتقة لأن يتم إعادة قيادة override واحد،أو بعض ،أوكل هذه الأعضاء، ا يوفر تفاصيل تنفيذ تكميلي أو يز خاص ا.توفر الفئات ا شتقة غالباً تفاصيل خاصة جموعة جزئية من الفئة القاعدية.مثلاً،الفئة القاعدية ال تعرف ا يوانات animalsستتضمن على وسيطات(واجهات) من أجل الأ اء ا شتركة، اسم الصنف باللاتينية،عدد الأرجل،وخاصيات وزجية أخرى تنتمي ميع ا يوانات.أما الفئة ا شتقة سوف سن ميزات الفئة القاعدية،ولكن فقط موعة جزئية من ا يوانات. كن لفئة الثدييات mammalأن تعرف مدة ا مل gestationلولادة الصغار،بينما فئة الطيور ا شتقة avian-derived class كن أن تعرف قطر البيضة.وكل من الثدييات mammalوالطيور avian ستبقى فظ خاصيات الاسمname، اسم الصنف species، وعدد الأرجل leg count من فئة ا يوان animalالقاعدية.حالة(نسخة) من الطيور avian ستكون حيوانanimal، ونسخة(حالة)من الثدييات mammalستكون حيوانanimal. ولكن ومهما يكن،فإن حالة من الطيور لن تكون ثدييات.وأيضاً، حالة عامة generic instance من حيوان animal كن أن يتم اعتبارها كـطيور فقط لو كانت بالأصل معرفة كطيور.

ح ولو كان كل من الفئة القاعدية وا شتقة لديهما علاقات،أو تفاصيل تنفيذ وال تكون خاصة privateبالفئة القاعدية ولن يتم عملها لأن تكون متاحة بالنسبة للفئة ا شتقة.فالفئة ا شتقة لا تعرف ح فيما إذا كانت الأعضاء ا اصة موجودة.الفئة القاعدية كن أن تتضمن أعضاء مية protectedوال ، وعلى الرغم من اختفاءها عن مستخدمي الفئة،فهي مرئية بالنسبة للفئة ا شتقة.وإن أي عضو معرف كعام Public الفئة القاعدية يكون متاح للفئة ا شتقة وأيضاً بالنسبة ميع مستخدمي الفئة القاعدية.(تعرف الفيجوال بيسك مستوى أخر مسمى صديق friend.والأعضاء ا علمة كصديق friend تكون متاحة لكل الكود نفس ا معassembly ، ولكن ليس خارج حدود كود ا مع.أما الأعضاء العامة Public كن أن يتم استخدامها بواسطة أي كود خارج نطاق ا مع ا عرف ا).

أمثلة الوراثة ا وجودة عملياً العا ا قيقي.فالساعة هي كائن قاعدي منه تشتق ساعة التنبيه(ا نبه).تعرض ساعة التنبيه واجهة عامة للساعة،وتضيف تنفيذ خاص ا بالنسبة للطرق وا اصيات،جدول وجدول دوري مشتق للعناصر الكيميائية.

تعدد الأشكال Polymorphism

ا بادئ ال تقد ها ح الآن كن أن يتم تنفيذها باستخدام لغات ال ة الإجرائية.على الرغم من أنك لاتستطيع عمل وراثة حقيقية لغة غ كائنية التوجهnon-OOP language. مثل لغة C،تستطيع اكا ا باستخدام حقول الإشارةflag :فإذا كان حقل إشارة flag field مسمى"نوع"type " فئة لغة غ كائنية التوجه – مثل تركيب وضعه إ "ثدييات"mammal ،"تستطيع ك استخدام حقول ثدييات نوعية mammal-specific fields .ددة.
يوجد طريقة أخرى اكاة هذه ا يزات،ولن تكون صعبة جداً.

تعدد الأشكال *Polymorphism* هي طيور تلفة اماًيع تعدد الأشكال"عدة ازج many forms." لأن الفئة ا شتقة كن أن وي نسخة(معادة القيادة overridden) خاصة ا من عضو لفئة قاعدية،إذا عاملت كائن الثدييات *mammal* مثل حيوان عام*animal* generic ،فيمكن أن يكون هناك بعض التشويش(الفوضى)فأي إصدار من الأعضاء سيتم استخدامه،إصدار ا يوان أم إصدار الثدييات.تعدد الأشكال يأخذ بع الاعتبار استكشاف كل هذا،وعلى هذا الأساس،بينما يكون برنا ك مشتغل، عل تعدد الأشكال من ا مكن لأي كود برنا ك معاملة حالة مشتقة derived instance وكأ ا حالة قاعديةbase instance. وهذا ما عل كتابة الكود شيء عظيم.فإذا كان لديك روت يتعامل مع كائنات ا يوان*animal*، بإمكانك ،رير له كائنات من نوع حيوان*animal*، أو ثدييات*mammal*، أو طيور *avian*،وسيبقى يعمل.هذا النوع من تعدد الأشكال يعرف تعدد أشكال فرعي *subtyping polymorphism*،ولكن من يهتم بالاسم مهما يكن.

تنوع أخر لتعدد الأشكال وهو "إعادة التعريف Overloading ." يسمح إعادة التعريف Overloading لطريقة فئة مفردة(إنس الفئات ا شتقة الآن)لأن يكون لديها عدة ازج،ولكن سيبقى اعتبارها كطريقة مفردة.مثلاً،إذا كان لديك كائن م ل مع الطريقة "طلاء"(وال ستغ لون ا ل)،تستطيع امتلاك طريقة طلاء واحدة وال تقبل لون وحيد(طلاء ا ل كله بلون واحد)،وطريقة طلاء أخرى تقبل لون (لون رئيسي زائد لون تشذيب(زخرفة)).عندما يتم إعادة قيادة هذه الطرق فئة مفردة، دد ا ترجم أي إصدار سيتم استدعاءه بالاستناد على البيانات ال تضمنها الاستدعاء للطريقة(ا عاملات النسبية ال ررها وعددها).

الواجهات والتنفيذ Interfaces and Implementation

يفرق(يز) تطوير ال ة كائنية التوجه ب التعريف العام لفئة(الكود ا كتوب لتنفيذ تلك الفئة)والاستخدام الناتج(ا اصل) الذاكرة لتلك الفئة ككائن. وهذا مشابه إ حدما كيف تفرق، مطعم،ب قائمة الطعام(الطبخ الذي تاره)،والطعام ا قيقي الذي يظهر على الطاولة.
.وصف بند على قائمة الطعام هو(إ حدما)واجهته*interface* (واجهة البند)،فهو يصف ماذا سيعرض الكائن ا قيقي بشكل عام من ناحية(أو من حيث)الطعم،والرائحة، وهكذا.
.الطريقة ا ستخدمة بواسطة العامل ا طبخ لصنع الطعام هي التنفيذ*implementation*، وهي كيفية ض الوجبة. كن أن توجد تنفيذات تلفة من قبل مطاعم تلفة من أجل نفس بند قائمة. الكائنات،التنفيذ*implementation* يكون في عن العرض العام،ففي مطعم ما، ض الطعام هو بامتنان في عن ا شهد أو ولا واحد سيأكل هناك(مطبخ ا طعم).
.الطعام الذي تستقبله من ا طبخ هو الكائنobject، ا الة (النسخة)*instance* إ قيقية ا تصفه تلك القائمة.العديد من الزبائن ا ائع كن أن يطلبوا نفس بند القائمة،وكل منهم سيستقبل نسخة متمايزة distinct instance من الطعام.

ال ة كائنية التوجه الفيجوال بيسك والدوت نتOOP in Visual Basic and .NET

مبدئياً،ال ة الكائنية التوجه ليست بذلك التعقيد حقيقة، ا أن كل من جنس البشر وا يتفاعلان مع كائنات العا ا قيقي وحالاته كل يوم،فمن السهولة كان أن تترسخ عقو م فكرة ال ة بالكائنات.ولكن ما مدى سهولة ربط مبادئ الكائن هذه بالكمبيوتر من خلال مترجم الفيجوال بيسك وإطار عمل الدوت نت؟هل كن عمل هذا بدون دورات أسبوعية؟إ أقول بالطبع هذا سهل.
سبب واحد عل الكائنات سهلة جداً الدوت نت هو أنه ب أن يكون كل شيء برنامج الدوت نت خاصتك جزء من كائن ما،وإذا كان كل شيء كذلك. و ا أن كل شيء الدوت نت ليس صعب كث اً، لأن تنفيذ الفيجوال بيسك للكائنات يس على التوازي مع الأفكار التصورية للكائنات.

الفئات Classes

تستخدم الفيجوال بيسك الفئات *classes*والتراكيب *structures*لتعريف الكائنات.وسأتكلم حول التراكيب فيما بعد هذا الفصل.تبدأ الكلمة ا جوزة Class بتعريف الفئة كما يلي:

```
Class [اسم الفئة]
    ' يتم وضع الكود ا اص بالفئة هنا
End Class
```

هذا هو معظم الفئة :الكلمة ا جوزةClass،واسم الفئة . يع الفئات تقيع فضاء أ اء ما(مناقشتها قليلاً الفصل الأول).بشكل افتراضي،تظهر فئاتك فضاء أ اء ما ا تسميته أبعد من مشروعك.تستطيع تغ هذا صفحات خاصيات ا شروع(لوضع فضاء الأ اء ا ستوى الأعلى top-level namespace معك assembly)وبالعبارة الأ اء Namespace (لتش لفضاءات الأ اء ا تعلقة من فضاء أ اء ا ستوى الأعلى).

```
Namespace GoodGuys
    Class Superhero
    End Class
End Namespace
```

إذا كان فضاء أ اء تطبيقك الافتراضي هو WindowsApplication1،فالفئة عينة الكود هذه سيتم تعريفها كمايلي:

WindowsApplication1.GoodGuys.Superhero

تستطيع إضافة أي عدد من الفئات لفضاء الأ اء.الفئات ال تستخدم نفس الاسم،ولكن تظهر فضاءات أ اء تلفة،هي غ مرتبطة.
تظهر أعضاء فئة ما ب العبارة Class وEnd Class. تستطيع أيضاً تقسيم تعريف فئة ضمن عدة ملفات لكود مصدري.فإذا ما عملت على فصل فئة ثل هذا، ب أن يتضمن على الأقل واحد من الأجزاء الكلمة ا جوزة Partial التعريف:

```
Partial Class Superhero
End Class
```

كما مع تعريف ا تغ ات،يتم تعريف الفئات باستخدام واحد من الكلمات ا جوزة دداث الوصول:عامPublic، خاصPrivate، مي Protected، صديق Friend، أو صديق مي Protected Friend.
يتم ميل مكتبات فئة إطار عمل الدوت نت(FCLs) NET Framework Class Libraries. ببساطة مع الفئات والكائنات،ويتم تعريفها كلها بالكث من البساطة بالكلمة ا جوزة البسيطة:"فئةClass ".

أعضاء الفئات Class Members

استدعاء فئتك Superhero لن نحها أي قوة خاصة إذا تضيف أي عضو *members* للفئة. يع أعضاء الفئات ب أن تظهر ب ا ددين Class وEnd Class.على الرغم من ذلك إذا ما استخدمت ا يزة Partial لفصل فئتك،تستطيع نثر sprinkle الأعضاء ب الأجزاء ا ختلفة للفئة بأي طريقة ترغب.
تستطيع تضم 11 نوع تلف من الأعضاء فئات الفيجوال بيسك. كن أن تعطيك كتب أو مستندات أخرى أرقام تلفة،ولكنها طئة،لو نظموا الأشياء على الأقل بالطريقة ال أقوم ا هنا:

حقول ا تغ ات Variable fields

ا تغ ات ذات القيمة او رجعية كن أن يتم إضافتها مباشرة إ فئتك كأعضاء من ا ستوى العلوي top-level members. وكأعضاء للفئات بالكامل،فإنه كن الوصول ا بواسطة أي كود معرف نفس الفئة،ومن ا تمل من قبل أي كود يستخدم تلك الفئة.يتم تعريف ا تغ ات باستخدام واحد من دداث الوصول access modifiers.

```
Class Superhero
        Public Name As String
        Protected TrueIdentity As String
    End Class
```

إن حقول ا تغ سهلة ومناسبة ليتم إضافتها إ الفئات،ا تغ ات العامة كن الوصول ا كما تريد،بدون أي ديدات،ح ولو رغبت بتحديد ال السماحية للحقل.على الرغم من أن ا قول لا تعمل مباشرةً مع يع ميزات الفيجوال بيسك،متضمنةً بعض ا يزات ا اصة بـلينكوLINQ. عندما تطفو مشاكل مثل هذه،تستطيع استخدام خاصية الأعضاء members بدلاً من أعضاء حقول ا تغ . سأقدم ا اصيات properties عدة فقرات.

حقول الثوابت Constant fields

تعرف الثوابت constants اماً مثل حقول ا تغ ات،ولكنه يتضمن الكلمة ا جوزة Const. كما مع الثوابت على مستوى الإجراءات ا لية local procedure-level constants، ب أن تسند قيمة للثابت مباشرةً الكود ا صدري،باستخدام ا رفية literals أوعينة سابات غ متغ ة.

```
Private Const BaseStrengthFactor As Integer = 1
```

العدادات Enumerations

تعرف العدادات قيم عددية صحيحة مرتبطة.حا ا تعرفها،تستطيع استخدامها كودك مثل القيم العددية الأخرى اماً.

```
Private Enum GeneralSuperPower
        Flight
        Strength
        Speed
        VisionRelated
        HearingRelated
        WaterRelated
        TemperatureRelated
        ToolsAndGadgets
        GreatCostume
    End Enum
```

كن للعدادات أيضاً أن يتم تعريفها على مستوى فضاء أ اء،خارج أي فئة صصة.

الطرق الفرعية Sub methods

تتضمن الفئات نوع من الطرق:الإجراءات الفرعية subsوالدوال functions. كل الكود ا نطقي logic code تطبيقك يظهر واحد من أنواع هذه الطرق أو properties خاصيات،لذلك لاتتعب نفسك بالبحث عن مثل هذا الكود عداد ما.تعمل الطرق الفرعية Subبعض ا نطق ا دد،بشكل اختياري،العمل على البيانات ا مررة ضمنها هي ا عاملات النسبية.

```
Public Sub DemonstrateMainPower(ByVal strengthFactor As Integer)
        ' ----- Logic code appears here.
    End Sub
```

الطريقةDemonstrateMainPower، كعضو عام لفئتك،تستطيع استدعاءه إما بواسطة الكود ضمن الفئة،أو بواسطة كود مرجعي(يش (الة من فئتك.وتتضمن هذه الطريقة وسيط مفرد، strengthFactor،والذي من خلاله استدعاءات هذه الطريقة ترسل فيه بيانات معاملات نسبية data arguments.

تستطيع ا روج من الطريقة أي وقت باستخدام العبارةReturn، أو العبارة الأقدم ال كانت مستخدمة قبل الدوت نت Exit Sub.

طرق الدوال Function methods

طرق الدوال Functionتشبه كثً طرق الإجراءات الفرعيةsub، ولكنها تدعم قيمة عائدةreturn value .إنك دد نوع البيانات للقيمة العائدة باستخدام الشرط Asعند اية تعريف الدالة. تستطيع إسناد القيمة ا عادة باستخدام العبارة return،أو بإسناد اسم الدالة مباشرةً ضمن الكود.

```
Public Function GetSecretIdentity(ByVal secretPassword As String) As String
        If (secretPassword = "Krypton") Then
            ' ----- I created a class field named
            ' TrueIdentity earlier.
            Return TrueIdentity
        Else
            GetSecretIdentity = "FORGET IT BAD GUY"
        End If
    End Function
```

إذا استخدمت ط الإسناد لاسم الدالة assignment-to-function-name style لإسناد القيمة الراجعةreturn value assignment، استخدام العبارة Exit Function للعودة إ الكود ا ستدعي أي وقت.

ا اصيات Properties

مع ا اصيات Propertiesفكرة ا قول fieldsوالطرقmethods. تستطيع إنشاء خاصيات للقراءة والكتابةread-write، أو للقراءة فقط read-only،أو للكتابة فقط write-only من خلال ا وصلات accessors "احصل Get"و"ضع Set ". الكود التا يعرف خاصية للكتابة فقط:

```
Public WriteOnly Property SecretIdentity() As String
        Set(ByVal value As String)
            TrueIdentity = value
        End Set
    End Property
```

التفاويض Delegates

تعرف التفاويض معاملات نسبية argumentsوتعود بقيم valuesللطرقmethod، وتغلفها(تصندق ا عاملات النسبية) كائن وحيد كل ما صها من معاملات.يتم استخدامها بشكل عام لدعم معا ة ا دث،راجع الإجراءات،والاستدعاءات الغ مباشرة لطرق الفئة.

```
Public Delegate Sub GenericPowerCall(ByVal strengthFactor As Integer)
```

ا أن التفاويض عامة إ حدما،فيتم تعريفها غالباً على مستوى فضاء الأ اء،خارج تعريف أي فئة.

الأحداث Events

إضافة أحداث لفئتك يتيح ستهلكي فئتك للتفاعل مع التغ ات والإجراءات(الأفعالactions)ال دث ضمن حالة فئة.التركيب syntaxا ستخدم لتعريف الأحداث يبدو مشابه كثً ا لتعريف طريقةmethod، ولكن يستخدم التركيب البديل التفاويض ا عرفة مسبقاً ليدل على بصمة(توقيع signature) ا دث.

```
' ----- Non-delegate definition.
Public Event PerformPower(ByVal strengthFactor As Integer)
    ' ----- Delegate definition.
    Public Event PerformPower As GenericPowerCall
```

التصاريح Declares

تتيح لك عبارة التصريح Declareاستدعاء كود معرف ملفات DLL خارجيةexternal، ومع أنه يعمل مع استدعاءات DLLويندوز دوت نت سابقة. تركيب التصريح يشابه إ حد بعيد التركيب ا ستخدم لتعريف الطرق.

```
Public Declare Function TalkToBadGuy Lib "evil.dll" (ByVal message As String) As String
```

حا ا يتم التعريف، كن أن يتم استخدام دالة أو تفرع مصرح خارجياً كودك وكأنه تعريف دالة أو تفرع دوت نت معد للاستخدام(جاهز للاستخدام built-in).يقوم إطار عمل الدوت نت بالكث من العمل خلف ا شهد لنقل البيانات جيئة وذهاباً ب برنا ك و مكتبة الربط الديناميكيDLL . الاهتمام الذي ب أن يتم أخذه عند التفاعل مع الكود "الغ مدار unmanaged " ا ارجي،خاصةً إذا كانت مكتبة الربط الديناميكي مسماة *evil.dll*.

الواجهات Interfaces

تتيح لك الواجهات Interfacesتعريف فئات رة abstract classes وبطريقة ما،قوالب فئةclass templates . كن أن يتم تعريف الواجهات على مستوى فضاء الأ اء،وعادةً هي كذلك.

الأنواع ا تداخلة Nested types

كن أن تتضمن الفئات فئات ثانوية (أو تراكيب) أخرى من أجل استخدامها ا اص الداخلي أو العام.إذا جعلت فئة مثل "فئة الابنchild " عامة.تستطيع العودة بنسخ(حالات) ذه الفئات لكتابة كود يستخدم فئة "آباء parent " كأ .

```
Class Superhero
    Class Superpower
    End Class
End Class
```

بإمكانك جعل الفئات متداخلة بأي عمق تريد،ولكن لا تتنحى جانباًdon’t go overboard. إنشاء فئات متعددة ضمن نفس فضاء الأ اء سيلاقي على الأغلب احتياجاتك بدون جعل الكود معقد بإفراط.ولكن هذه فكر فقط،افعل ما تريد.فهو كودك بعد كل شيء.

إضافة تنوع variety يل من الأعضاء لفئة ما فيه الكث من ا تعة.تستطيع إضافة أعضاء فئة بأي تنوع ،بأي ترتيب،وبأي كمية.

أعضاء الفئة ا تشاركة Shared Class Members

عادةً،الكائنات(حالات فئة) هي جشعة وأنانية،فهي تريد أن فظ كل شيء لنفسها ولا تشارك به الكائنات الأخرى.و ذا السبب كل حالة من فئة تعمل على إنشاءها لديها إصدار خاص ا من عناصر البيانات يتم تعريفه كأعضاء فئة.وحـ طرق و خاصيات فئة الـ نح ا ظهر لأن يكون متميز بالنسبة لكل حالة فئة.

تتضمن الفيجوال بيسك الكلمة ا جوزة Shared تصميم فئا ا.وهذه الكلمة ا جوزة Shared كن أن يتم تطبيقها قل متغ variable field،طريقة التفرعsub، طريقة الدالةfunction، و خاصية propertyلأعضاء فئتك.بعد التعريف،الأعضاء ا تشاركة *shared member* كن أن يتم استخدامها دون ا اجة لإنشاء حالة *instance*من تلك الفئة.إنك تشـ إ هذه الأعضاء ا تشاركة باستخدام فقط اسم الفئة واسم العضو.

```
Class ClassWithSharedValue
    Public Shared TheSharedValue As Integer
End Class
'...later, in some other code...
ClassWithSharedValue.TheSharedValue = 10
```

الأعضاء ا تشاركة هي حرفياً"متشاركةshared " كل حالات instancesفئاتك،وإذا كانت الفئة عامة تصبح متشاركة بكود خارج نطاق الفئة أيضاً. ا أ لا ا تاج إ حالة كائن ما،فهي أيضاً دودة ذه ا صادر فقط الـ لاتتطلب حالة كائن.وهذا يع أن الطرق ا تشاركة لا كن ا أن تصل إ حقول ا تغ ات الغ متشاركة لنفس الكائن.أي أعضاء فئة والـ يتم تعليمها كـ متشاركة تعرف بأعضاء حالة *instance members* .

الأعضاء ا عاد تعريفها وا عاملات النسبية الاختيارية Overloaded Members and Optional Arguments

دث إعادة تعريف طريقة methodبا اق الكلمة ا جوزة Overloadsلكل عضو معاد قيادته.

```
Class House
    Public Overloads Sub PaintHouse()
        ' ----- Use the same color(s) as before.
    End Sub
    Public Overloads Sub PaintHouse(ByVal baseColor As Color)
        ' ----- Paint the house a solid color.
    End Sub
    Public Overloads Sub PaintHouse(ByVal baseColor As Color,ByVal trimColor As Color)
        ' ----- Paint using a main and a trim color.
    End Sub
    Public Overloads Sub PaintHouse(ByVal baseColor As Color,ByVal coats As Integer)
        ' ----- Possibly paint with many coats, of paint
        ' that is, not of fabric.
    End Sub
End Class
```

عندما تستدعي الطريقةPaintHouse ، ب أن رر ا عاملات النسبية الـ تطابق واحد من الإصدارات ا عاد قياد ا. دد الفيجوال بيسك أي إصدار سيتم استخدامه بالاعتماد على توقيع signature ا عاملات النسبية.إذا مررت نوع أو عدد من ا عاملات النسبية خاطئ،فإن الـ نامج س فض التر ة.

سيبدو عضوين معاد قياد م هذه الفئة كالتا ،فيما عدا ا عامل النسـ الثا coats :

```
Public Overloads Sub PaintHouse(ByVal whichColor As Color)
Public Overloads Sub PaintHouse(ByVal baseColor As Color, ByVal coats As Integer)
```

بدلاً من تعريف طريقتـ متمايزتـ ،تستطيع د هما طريقة مفردة،ومعامل نسـ اختياري *optional argument* دد بالنسبة للوسيط coats.

```
Public Overloads Sub PaintHouse(ByVal baseColor As Color, Optional ByVal coats As Integer
= 1)
    End Sub
```

الكلمة ا جوزة Optional كن أن يتم استخدامها على أي عدد من الوسيطات،ولكن لا كن أن تظهر الوسيطات غ الاختيارية بعدها،فيجب أن تكون ا عاملات النسبية الاختيارية دائماً أخر القائمة.على الرغم من أن الكود ا ستدعي كن أن لا رر قيمة للوسيط coats،ولكن ما تزال الدوت نت تتطلب أن يستقبل كل وسيط معامل نسـ .وهكذا،كل معامل نسـ اختياري يتضمن قيمة افتراضية *default value* باستخدام الإسناد البسيط ضمن تعريف الوسيط. ا عامل النسـ الاختياري coatsيستخدم القيمة الافتراضية 1من خلال الشرط1 = .

الوراثةInheritance

تدعم الفيجوال بيسك الوراثة،ربط فئة علاقة أجداد(أباء)-أحفاد(سلالةancestor-descendant).لتنفيذ الوراثة،عرف الفئة القاعدية base class،ومن أضف الفئة ا شتقة derived classباستخدام الكلمة ا جوزة Inherits.

```
Class Animal
    ' ----- Animal class members go here.
End Class
Class Mammal
    Inherits Animal
    ' ----- All members of Animal are automatically
    ' part of Mammal. Add additional Mammal
    ' features here.
End Class
```

العبارة Inherits ب أن تظهر عند بداية تعريف الفئة،وقبل أي تعريف لأعضاء الفئة.و ب أن تتضمن نفس اسم الفئة القاعدية بالضبط.إذا فصلت فئتك ا شتقة باستخدام الكلمة ا جوزةPartial ، تاج لاستخدام العبارة Inherits واحد من القطع فقط.و ا أن الفئة ا شتقة كن أن تستخدم فئة قاعدية مفردة فقط،فيتم ديد استخدام العبارة Inheritsمرة فقط كل فئة.(كن أن يتم استخدام الفئة القاعدية عدة فئات مشتقة تلفة،و كن أن يتم استخدام الفئة ا شتقة إ حد أبعد كفئة قاعدية بالنسبة لفئات مشتقة أخرى.)

الفئات ا شتقة ترث بشكل آ يع الأعضاء ا عرفة للفئة القاعدية.إذا ما احتاجت فئة مشتقة توفر صص وظيفي خاص لعضو معرف الفئة القاعدية،فإ ا تعيد قيادة overridesذلك العضو.وهذا يتم طوت ☹1) ب أن تسمح الفئة القاعدية لعضو منها لأن يتم إعادة قيادته بالكلمة ا جوزة Overridable،و(2) ب أن توفر الفئة ا شتقة كود إعادة القيادة باستخدام الكلمة ا جوزة Overrides.

```
Class Animal
        Public Overridable Sub Speak()
            MsgBox("Grrrr.")
        End Sub
    End Class
    Class Canine
        Inherits Animal
        Public Overrides Sub Speak()
            MsgBox("Bark.")
        End Sub
    End Class
```

إن أي فئة تشتق من الفئة حيوان Animalتستطيع الآن توفر كودها ا اص او خصص من أجل الطريقة Speak.ولكن نفس الشيء صحيح من اجل الفئات ا شتقة من فئة الكلاب" Canine "،يتم رير الكلمة ا جوزة لكل جيل.إذا احتجت لوقف هذه ا واصفة attributeعند جيل مع ، استخدم الكلمة ا جوزةNotOverridable .تكون هذه الكلمة ا جوزة قة عندما يتم استخدامها فقط الفئة ا شتقة ا أن أعضاء الفئات القاعدية هي غ قابلة لإعادة للقيادة nonoverridableبشكل افتراضي نفس الفئة القاعدية فالأصل وضع الكلمة ا جوزة Overridable لتتمكن الفئة ا شتقة من إعادة قيادة أعضاء الفئة القاعدية.

```
Class Canine
        Inherits Animal
        Public NotOverridable Overrides Sub Speak()
            MsgBox("Bark.")
        End Sub
    End Class
```

أحياناً عندما يكون من غ مكن كتابة طريقة عامة حقاً الفئة القاعدية،وتريد أن تطلب من كل تعريف لفئة مشتقة أن يكون ا إصدار خاص ا من الطريقة.باستخدام الكلمة ا جوزة MustOverride تعريف عضو لفئة قاعدية كّن هذا الطلب.

```
Class Animal
        Public MustOverride Sub DefenseTactic()
End Class
```

الأعضاء ا علمة كـ MustOverrideلاتتضمن تنفيذ كود خاص ا، ا أ ا لن تستخدم.(أيضا لاحظ أن الطريقة DefenseTacticليس لديها End Sub).بسبب عدم وجود كود مصاحب ذا العضو،فإن فئة ا يوان بالكامل فيها نقصdeficiency .

إذا أنشئت حالة من فئة ا يوان Animal،واستدعيت طريقتهاDefenseTactic، سينتج هلع ضمن تطبيقك.وهكذا،ليس من ا مكن إنشاء حالات من الفئات ال توي أعضاء" ب قياد ا MustOverride ."للتخلص من هذا التحديد،يتم وسم الفئة أيضاً بالكلمة ا جوزةMustInherit .

```
Public MustInherit Class Animal
        Public MustOverride Sub DefenseTactic()
    End Class
```

لن يكن من ا مكن إنشاء حالة من فئة ا يوان Animalمباشرة،على الرغم من أنك تستطيع اشتقاق فئات منها،وإنشاء حالات من هذه الفئات.وأيضاً،تستطيع إنشاء متغ حيوانAnimal (من النوع ا رجعيreference type) وتسند حالة مشتقة لفئة ا يوان Animal-derived class له(تغ ا يوان).

```
Dim zooMember As Animal
    Dim monkey As New Simian ' Simian is derived from Animal
zooMember = monkey
```

مثل هذا الكود لايبدو عادلاً حقاً بالنسبة للفئة القاعدية.أءِ أ، ا عرفت كل ا تطلبات وهرية بالنسبة للفئات ا شتقة، ولكنها لا صل على أي اعتماد(رصيد) ا أنه لا كن عمل نسخة(حالة) مباشرة منها.ولكن توجد طريقة لتسيطر الفئة القاعدية على قدرها ا اص،لتأخذ كل ا د لنفسها.فهي تعمل هذا بالكلمة ا جوزة NotInheritable.

```
NotInheritable Class Animal
    End Class
```

الطريقة الوحيدة لاستخدام فئة غ قابلة للوراثة NotInheritableهو إنشاء حالة ما منها،لا كنك استخدامها كفئة قاعدية لفئة مشتقة أخرى.(إذا فئتك الغ قابلة للوراثة noninheritable تحتوي على أعضاء متشاركة، كن أن يتم الوصول ا بدون ا اجة لإنشاء حالة ما.)

يورثInherits ، ب التوريثMustInherit، غ قابل للوراثةNotInheritable، إعادة القيادةOverrides، قابل للقيادةOverridable، غ قابل للقيادة NotOverridable،هذه ليست جدة الفيجوال بيسك بعد الآن.وسيبقى يوجد واحد آخر ذه الكلمات ا جوزة الـ لاتضاهىinimitable : التظليل Shadows.عندما تعيد قيادة عضو فئة قاعدية، ب أن يستخدم الكود ا ديد تعريف يث يكون موافق لواحد من ما هو مزود الفئة القاعدية.هذا كل شيء،لو أعدت إعادة قيادة طريقة دالة functionمع معامل نسيب من نوع السلسلة ا رفية Stringوالكود الراجع من نوع العددي الصحيح Integer return code،فيجب أن يستخدم كود إعادة القيادة نفس ذلك التوقيع. الأعضاء ا ظللة ليس لديها مثل هذه ا تطلبات.تطابق الأعضاء ا ظللة بند الفئة القاعدية"بالاسم فقط".كل شيء آخر يرجع إ الاغتصاب(الانتزاعgrabs).تستطيع ح تغي نوع العضو.إذا كان لديك طريقة تفرع مسماة PeanutButter الفئة القاعدية،تستطيع تظليله الفئة ا شتقة قل متغ (أو ثابتconstant، أو عدادenumeration، أو فئة متداخلة nested class)ومسمى أيضاً PeanutButter.

```
Class Food
        Public Sub PeanutButter()
        End Sub
    End Class
    Class Snack
        Inherits Food
        Public Shadows PeanutButter As String
        ' Hey, it's not even a "Sub"
    End Class
```

بدون الكلمة ا جوزة Shadows الفئةSnack، سيحدث خطأ وقت التر ة.

إنشاء حالات من الفئات Creating Instances of Classes

ا طوة الأ :تصميم الفئات designing classes. ا طوة الثانية:اشتقاق الفئاتderiving classes. ا طوة الثالثة:إنشاء حالات فئة creating class instances.تستخدم الفيجوال بيسك الكلمة ا جوزة Newلإنشاء حالات instancesمن فئتك ا خصصة.

```
Dim myPet As Animal = New Animal
        ' ----- Or...
Dim myPet As New Animal
        ' ----- Or...
Dim myPet As Animal
        myPet = New Animal
```

ا الة كن أن يتم استخدامها مثل أي متغ حالة للدوت نت.و كن الوصول للأعضاء باستخدام الرمز "نقطة dot "

```
myPet.Name = "Fido"
```

تستطيع أيضاً(للسبب ما) رير متغ ات حالة ب تغ ا م القاعدية وا شتقة.

```
Dim myPet As Animal
        Dim myDog As Canine
        myDog = New Canine
        myDog.Name = "Fido"
        myPet = myDog ' Since Canine derives from Animal
        MsgBox(myPet.Name) ' Displays "Fido"
```

إذا وضعت خيار التدقيق Option Strict لـ(On فعال)،سيكون هناك حدود على قدرتك للتحويل ب الأنواع،وخاصة ويلات التضييق narrowing conversions (حيث نوع البيانات ا صدرية لن تتوافق دائماً ا تغ ا دف). مثل هذه ا الات،يتوجب عليك استخدام الدالة CType (أو واحد من الدوال ا شا ة ا زودة من قبل الفيجوال بيسك والدوت نت)لتتمك التحويل.

```
myDog = CType(myPet, Canine)
```

الإشارة إ حالات فئة هو ببساطة مسألة الإشارة إ متغ أو كائن وي ا الة.وهذا صحيح من أجل الكود الذي يستخدم حالة من خارج الفئة نفسها.أما من أجل الكود ضمن فئتك(مثل واحدة من طرق فئتك)فإنك تش إ أعضاء حالتك وكأ م متغ ات لية(بدون قيد qualification)أو استخدام الكلمة ا جوزة ا اصة Me.

```
Class Animal
        Public Name As String
        Public Sub DisplayName()
            ' ----- Either of these lines will work.
            MsgBox(Name)
            MsgBox(Me.Name)
        End Sub
    End Class
```

كلمة جوزة مشا ةMyClass، عادة تتصرف مثل الكلمة ا جوزةMe، ولكن لديها بعض التخصص الوظيفي ا ختلف عندما يتم زين حالة فئة متغ من نوع فئة تلف(قاعدية أو مشتقة).إذا أنشئت حالة من فئة "الكلابCanine "،ولكن خزنتها متغ لفئة حيوانAnimal ، فا رجعيات باستخدام الكلمة ا جوزة Meسترتكز على كود"الكلاب Canine "،بينما مرجعيات لـ MyClass سترتكز على كود حيوانAnimal. ولن استخدم مشروع ا كتبةMyClass، ومن أجل معظم الاستخدامات البسيطة الات فئة،لن تستخدمها أيضاً.ولكن يوجد أحيان يكون من ا ام التميز ب كود الفئة القاعدية وا شتقة،وهذه هي الطريقة لفعل هذا.

تش الكلمة ا جوزة MyBaseلعناصر الفئة القاعدية وال منها اشتقاق الفئة ا الية.وهي تش للفئة الأقرب فقط.،إذا كان لديك فئة مسماة Class5والClass4 تشتق من Class4،والClass3 بدورها تشتق من Class3،والClass2 تشتق أيضا من Class2،والClass1 أيضاً تشتق من Class1،والنهاية تشتق من System.Object . بالإشارة إ MyBase كود الفئة Class5فهو يش إ Class4 . حسناً،هذا على الأغلب صحيح.إذا حاولت استخدام MyBase.MemberName،و MemberNameغ موجودة الفئةClass4 ، سترجع MyBaseوتبحث خلال كومة الفئات ح التعريف الأقرب ل MemberName.

```
Class Animal
    Public Overridable Sub ObtainLicense()
        ' ----- Perform Animal-specific licensing code.
    End Sub
End Class
Class Canine
    Inherits Animal
    Public Overrides Sub ObtainLicense()
        ' ----- Perform Canine-specific licensing code, then...
        MyBase.ObtainLicense() ' Calls code from Animal class
    End Sub
End Class
```

ا شيدات وا هدمات Constructors and Destructors

الات فئة عمر ما:البداية،وقت بدء الفعالية،وأخ اً، ا اية. دث بداية عمر كائن ما من خلال مشيد ما*constructor*، و ظاته الأخ ة يتم إملاءها dictatedبواسطة ا هدم *destructor*قبل ريرها عمليات تفريغ مصادر النظام garbage collection للدوت نت.

تتضمن كل فئة على الأقل مشيد واحد،إما صريح explicitأو ضم implicit. إذا توفر مشيد،فإن الدوت نت على الأقل سيصنع مشيد أصغري على مستوى النشاطاتminimal constructor-level activities، مثل حجز reservingمساحة ذاكرة memory space لكل حالة حقل متغ instance variable fieldلفئتك.إذا أردت من فئة أن يكون لديها أي منطق أخر وقت البدء،عليك تزويده من خلال مشيد صريح.

ا شيدات الفيجوال بيسك هي طرق فرعية مع الاسم ا شيد بدون معاملات نسبية يتصرف كمشيد افتراضي*default constructor*، يتم استدعاءه بشكل افتراضي م كانت هناك حاجة لفئة.

```
Class Animal
    Public Name As String
    Public Sub New()
        ' ----- Every animal must have some name.
        Name = "John Doe of the Jungle"
    End Sub
End Class
```

بدون ا شيد constructor، حالة جديدة من حيوان Animalلن يكون ا أي اسم مسند قل الاسمName. و ا أن متغ ات السلاسل ا رفية Stringهي أنواع مرجعيةreference types ، بالتا سيكون للاسم قيمة أولية"لاشيءNothing ."وليس سهل الاستخدام كث اً. نحك ا شيد الافتراضي فرصة لتوف على الأقل البيانات وا نطق الأصغري ا طلوب الة جديدة.

تستطيع توف مشيدات صصة إضافية بإضافة طرق "جديدة New " أكثر،كل منها بتواقيع معاملات نسبية argument signature تلفة.

```
Class Animal
    Public Name As String
```

```
    Public Sub New()
        ' ----- Every animal must have some name.
        Name = "John Doe of the Jungle"
    End Sub
    Public Sub New(ByVal startingName As String)
        ' ----- Use the caller-supplied name.
        Name = startingName
    End Sub
    Public Sub New(ByVal startingCode As Integer)
        ' ----- Build a name from a numeric code.
        Name = "Animal Number " & CStr(startingCode)
    End Sub
End Class
```

يوضح الكود التا كل مشيد:

```
MsgBox((New Animal).Name)
        ' Displays "John Doe of the Jungle"
        MsgBox((New Animal("Fido")).Name)
        ' Displays "Fido"
        MsgBox((New Animal(5)).Name)
        ' Displays "Animal Number 5"
```

تستطيع إجبار مستهلك فئتك على استخدام مشيد صص وذلك بإخراج ا شيد الافتراضي من تعريف الفئة.وأيضاً،إذا كنت تشتق فئتك من أي شيء غ System.Object،فهذه عادةً فكرة جيدة لاستدعاء مشيد الفئة القاعدية كأول سطر شيدك ا شتق،على الرغم من أن ا شيد الافتراضي الفئة القاعدية سيتم استدعاءه بشكل افتراضي.

```
Class Canine
    Inherits Animal
    Public Sub New()
        MyBase.New() ' Calls Animal.New( )
        ' ----- Now add other code.
    End Sub
End Class
```

قتل Killingحالة الفئة ليس بالسهولة ال قد تبدو عليه.عندما تنشئ حالات فئة لية طرقك،فإنه سيتم تدم ها بشكل آ عند ا روج من تلك الطريقة إذا تكن قد أسندت ا الة تغ خارج الطريقة.إذا أنشئت حالة ما طريقة وأسند ا ا عضو فئة،فا ا ستخلد عضو الفئة لعمر الفئة،حـ ولو قد ا روج من تلك الطريقة ا نشئة.

ولكن لنفكر فقط با الات ا لية حالياً.يتم تدم حالة عند ا روج من الروتـ .تستطيع أيضاً تدم حالة ما مباشرة بوضع متغ ها إ لاشيء.

myDog = Nothing

وضع ا تغ إ حالة نسخة جديدة سيدمر أي حالة سابقة زينها ذلك ا تغ .

```
myDog = New Canine
myDog.Name = "Fido"
myDog = New Canine ' Sorry Fido, you're gone
```

عندما يتم تدم كائن ما.تستدعي الدوت نت طريقة خاصة مسماةFinalize،إذا حضرت فهي لتقوم بأي تنظيف ائي قبل إزالة ا الة من الذاكرة. يتم تنفيذ Finalizeكطريقة مية Protectedلفئة System.Object القاعدية،لاستخدامها يتوجب عليك إعادة قيادة overrideهذه الطريقة فئتك.

```
Class Animal
    Protected Overrides Sub Finalize()
        ' ----- Cleanup code goes here. Be sure to call the
        ' base class's Finalize method.
        MyBase.Finalize()
    End Sub
End Class
```

ا شكلة أن الدوت نت تسيطر على استدعاء الطريقةFinalize، فهي جزء من عملية(معا ة)تفريغ مصادر النظامgarbage collection. فإطار العمل لا ينظف باستمرار نفاياته(بياناته العد ة النفع).فهي مثل ا دمة(خدمة البلدية) لك،فيتم التقاطها بواسطة شاحنة القمامة مرة واحدة كل فترة.وحـ ذلك ا تعمل على وضع النفايات هناك،تفسد،تتعفن،تتحلل،وليس لديها استدعاء للطريقة.من أجل معظم الكائنات،هذا ليس با شكلة الكبـ ة،فمن سيهتم إذا ما رير ذاكرة من سلسلة حرفية الآن أو بعد 30ثانية من الآن.ولكن يوجد أوقات يصبح من ا ام رير مصادر مكتسبة بسرعة قدر الإمكان. توجد طريقة صوص هذه ا شكلة.أحدها إضافة طريقة تنظيف cleanup مفصولة لفئتك يث تتوقع أي كود يستخدم فئتك لاستدعائه.وهذا سيعمل حـ ينسى كود ما استدعاء الطريقة.(وهكذا عليك أيضاً استدعاء هذا الروتـ من ا هدم.) الطريقة الثانية مشا ة،ولكنها تستخدم الواجهة ا زودة من قبل إطار العمل تدعى IDisposable.(سأ دث حول الواجهات لاحقاً).

```
Class Animal
    Implements IDisposable
    Protected Overrides Sub Finalize()
        ' ----- Cleanup code goes here. Be sure to call the
        ' base class's Finalize method.
        MyBase.Finalize()
    End Sub
    Public Overloads Sub Dispose() Implements IDisposable.Dispose
        ' ----- Put cleanup code here. Also make these calls.
        MyBase.Dispose() ' Only if base class is disposable.
        System.GC.SuppressFinalize(Me)
    End Sub
End Class
```

الطريقة SuppressFinalize مفرغ مصادر النظام garbage collector، "لاتستدع ا نهي Finalize، لقد قمت بتنظيف كل شيء سابقاً."أي كود يستخدم فئتك سيحتاج لاستدعاء طريقته Dispose لعمل تنظيف مباشر للمصادر.لذلك،لا تلف كثً اً عن الطريقة الأو ال دثت حو ا،ولكنها تعمل القليل من الأشياء ا قياسية (ا عيارية) .أيضاً،إ ا كن استخدام عبارة الفيجوال بيسك Using. ومقطع العبارة هذه توفر طريقة تركيبية لتنظيف ا صادر.

```
Using myPet As New Animal
        ' ----- Code here uses myPet.
    End Using
    ' ----- At this point, myPet is destroyed, and Dispose is
    ' called automatically by the End Using statement.
```

الواجهات Interfaces

الكلمات ا جوزة MustOverride و MustInherit الفئات ا شتقة على تنفيذ أعضاء خاصة من الفئة القاعدية.ولكن ماذا لو أردت من الفئة ا شتقة أن تنفذ يع أعضاء الفئة القاعدية؟ كن أن تستخدم MustOverride انب كل طريقة وخاصية،ولكن طريقة أفضل وهي استخدام الواجهة *interface*.تعرف الواجهات Interfaces فئات ردة *abstract classes*، فئات مؤلفة من التعريفات definitions فقط،بدون التنفيذ implementation.(تصطفي ال ة كائنية التوجه الرغبة ال دد أن فئة ما ح مع إشارة MustOverride واحدة فقط هي فئة ردة abstract class).تعمل الواجهات على إنشاء عقد *contract*، أو اتفاقية agreement تنفذ فئة أو تركيب يوافق(يلائم) التنفيذ.

تبدأ عبارة الواجهة عملية التعريف بـ interface. بالنسبة للعرف(التقليد By convention)،كل أ اء الواجهة تبدأ رف كب .

```
Interface IBuilding
    Function FloorArea() As Double
    Sub AlterExterior()
End Interface
```

كما ترى هنا،التركيب نوعا ما نسخة مبسطة عن تركيب تعريف الفئة. يع أعضاء الواجهة هي بشكل آ عامة،لذلك لايتم تضم دات الوصول access modifiers. ما هو مطلوب فقط سطر التعريف لكل عضو ا أنه ليس هناك تنفيذ.بالإضافة إ طرق الدوال والتفرع، كن أن تتضمن أعضاء الواجهات ا اصيات properties،الأحداث events ، واجهات interfaces أخرى،فئات classes،وتراكيب structures، و كن أيضاً أن يتم اشتقاق الواجهات interfaces من واجهات أخرى interfaces(باستخدام الكلمة ا جوزة Inherits)وبشكل آ يتم تضم كل أعضاء الواجهة القاعدية ا شتقة.

إنك تربط الواجهات بالفئة باستخدام الكلمة ا جوزة Implements. وهذه الكلمة ا جوزة يتم استخدامها فيما بعد لتش إ أي عضو فئة يعرف أي عضو واجهة.

```
Class House
    Implements IBuilding
    Public Function FloorArea() As Double Implements IBuilding.FloorArea
        ' ----- Add implementation here.
    End Function
    Public Sub PaintHouse() Implements IBuilding.AlterExterior
        ' ----- Add implementation here.
    End Sub
End Class
```

تنفيذات الفئة لأعضاء واجهة غ مطلوبة فظ اسم عضو الواجهة الأصلي(على الرغم من أن توقيع ا عامل النس ب أن توافق ما هو موجود الواجهة). عينة الكود، FloorArea حفظ اسم العضوا كافئ له الواجهة،ولكن العضو AlterExterior تنفيذه باستخدام الطريقة PaintHouse.وهذا ما عل بعض الكود ا متع كن.

```
Dim someHouse As New House
    Dim someBuilding As IBuilding
    someBuilding = someHouse
    someBuilding.AlterExterior() ' Calls someHouse.PaintHouse( )
```

كن أن ترث الفئات من فئة قاعدية واحدة فقط،ولكن لايوجد حدود على عدد الواجهات ال تستطيع أن تنفذها فئة ما.

```
Class House
    Implements IBuilding, IDisposable
```

وأيضاً،عضو فئة مفرد كن أن ينفذ أعضاء متعددة لواجهة.

```
Public Sub PaintHouse() Implements IBuilding.AlterExterior, IContractor.DoWork
    End Sub
```

لذلك، اذا استخدام الواجهات؟توفر الواجهات طريقة شاملة genericللوصول إ التخصص الوظيفي ا شترك،ح ب الأعضاء ال ليس لديها شيء مشترك.الفئات ا سماء Animal ، House وSuperhero، من ا تمل أن لايكون لديها شيء مشترك من حيث ا نطقlogic ،ولكن كن أن تاج يعها طريقة مناسبة(راسخة) لتنظيف مصادرها.فلو كان كل منها ينفذ الواجهةIDisposable ، فهي صل على تلك الإمكانية بدون ا اجة للاشتقاق من فئة قاعدية مشتركة ما.

الوحدات ال ية والتراكيبModules and Structures

بالإضافة إ الفئات،توفر الفيجوال بيسك ميزة تعريف كائن :التراكيب structuresوالوحدات ال يةmodules. على الرغم من أن ما أ اء تلفة عن الفئة،فهما يتصرفان بكثر من الشبه للفئات،ولكن مع ميزات تلفة كنة أو معطلة.

توفر الوحدات ال ية مكان لتضم كود عام شامل وقيم بيانات تطبيقك أو معك. يع أعضاء وحدة بر ية متشاركة. ا قيقة،تتصرف الوحدة ال ية مثل الفئة اماً مع الكلمة ا جوزة IShared ضافة لكل عضو اماً،ويبقى اختلاف وحيد رئيسي:لايوجد علاقات وراثة مسموح ا الوحدات ال ية.فلن تستطيع إنشاء وحدة بر ية مشتقة من فئة قاعدية أو وحدة بر ية .ولا ح كن استخدام وحدة بر ية ما كوحدة قاعدية لأي نوع أخر. الوحدات ال ية نقلها من اصدارات دوت نت سابقة للفيجوال بيسك،وال ضمنت "الوحدات ال يةModules " لكل الكود الغ منتمي للنماذج(الفورمات).

```
Friend Module GenericDataAndCode
    ' ----- Application-global constant.
    Public Const AllDigits As String = "0123456789"
    ' ----- Application-global function.
    Public Function GetEmbeddedDigits(ByVal sourceString As String) As String
    End Function
End Module
```

لاتستطيع إنشاء حالة ما من الوحدة ال ية.وكما مع الفئات،تظهر الوحدات ال ية سياق فضاء أ اء ما.ولا تشبه الفئة بالأعضاء ا تشاركة،لا تاج لتخصيص اسم الوحدة ال ية لاستخدام أعضاء الوحدة ال ية. يع أعضاء الوحدات ال ية تتصرف كمتغ ات وطرق شاملة،و كن أن يتم استخدامها مباشرةً أي كود أخر تطبيقك بدون شروط (مؤهلاتqualification)أخرى.(تستطيع تقييد عضو وحدة بر ية لأن يستخدم فقط الوحدة ال ية بالتصريح عنه كخاصPrivate)

التراكيب *Structures* أكثر شبهاً للفئات من الوحدات ال يةmodules. تنفذ الفئات أنواع مرجعية،ولكن تنفذ التراكيب أنواع ذات قيمة. يع التراكيب تشتق من System.ValueType (والذي بدوره يشتق منSystem.Object).وهي تتصرف مثل أنواع البيانات ذات القيمة ا وهرية للفيجوال بيسك،مثل العددي الصحيحInteger ،وتستطيع إنشاء حالات من التراكيب باستخدام نفس التركيب ا syntax ستخدم إنشاء حالات الفئة.ومهما يكن،لاتستطيع استخدام التركيب كقاعدة(تركيب قاعدي)من أجل تركيب أخر مشتق.وعلى الرغم من هذا تستطيع تضم مشيد constructor تركيبك،أما ا هدم ةdestructors مدعوم.

بسبب الطريقة ال يتم فيها زين واستخدام التراكيب الدوت نت،فهي مناسبة جيداً لأنواع البيانات البسيطة.تستطيع تضم أي عدد من الأعضاء تركيبك،ولكن من الأفضل حفظ الأشياء بسيطة.

الطرق ا زئية Partial Methods

سابقاً الفصل كتبت حول الفئات ا زئية،القدرة على تقسيم فئة ضمن عدة ملفات.الفئات ا زئية هي بشكل خاص معروفة الكود ا نشأ بواسطة مولدات الكودcode generators .

الفيجوال أستوديو،جزئياً،هي مولد كود،فعندما تسحب وتسقط أدوات على وذجك،فهي تولد كود لك فئة النموذج ا زئيpartial Form class . مثل هذه الفئات،الفئات ا زئية لديها مؤلفان:ا ولد الآ وأنت.

الطرق ا زئية*Partial methods* ،جديدة، الفيجوال بيسك2008،يتم استخدامها أيضاً بواسطة مولدات الكود،على الرغم من أنك حر توظيفها بنفسك.فهي مفيدة بشكل خاص عند تريد فئة مولدة آلياً أن نح مؤلفها الثانوي(أنت) القدرة على توف بعض ا نطق الاختياري والذي سيحسن ا نطق ا ولد بشكل آ .

كن أن تكون الطرق ا زئية أكثر دقةً بدعو ا"الطرق الاختيارية"، ا أن لديك ا يار بتنفيذها أو لا.

للطرق ا زئية جزئي ☹(1)النصف غ ا نفذ unimplemented half،و(2)النصف ا نفذ implemented halfالاختياريoptional. يظهر كلا النصف قسم تلف للفئة ا زئية.الطريقة ا زئية لايتم تقسيمها ب الفئة ا شتقة والقاعدية،فلا علاقة ا بالوراثة.

النصف غ ا نفذ للطريقة ا زئية يبدو كتعريف طريقة تفرغ فارغة،ولكن الكلمة ا جوزة Partialا ضافة.

```
Partial Private Sub ImplementIfYouDare()
    End Sub
```

الطرق ا زئية ب أن تكون دائماً طرق تفرعsub،لن تكون أبداً دوالfunctions،و ب أن يتم التصريح عنها كخاصةPrivate.إذا زودت أي وسيطات، ب أن تسبق بالكلمة ا جوزة ByVal،وليس بالكلمة ا جوزةByRef .

النصف ا نفذ implemented half يبدو معروفاً،ما عدا النقص(العوز) (السابقة"جزئيPartial" ولكن من ا ؤكد يبدو جيد مع الكود ا قيقي ب فكيه jaws.

```
Private Sub ImplementIfYouDare()
        MsgBox("I did it, so there.")
    End Sub
```

لذلك،ما هو التعامل الكب مع هذه الطرق ا زئية؟من ا تمل ليس الك ،ولكن إلقاء نظرة على مثال ما كن أن يساعد.دعنا نعود ياتنا،فئة ا يوان Animal،هذه ا رة مع الطريقة ا زئية ا ضمنة.لنبدأ مع ا انب ا ولد بشكل آ auto-generated side للعا .

```
Partial Class Animal
    Public Sub Move()
        ' ----- Interesting movement code, then...
        MoveSideEffects()
    End Sub
    Partial Private Sub MoveSideEffects()
    End Sub
End Class
```

بعض الأحيان عندما يتنقل moves حيوان،فلديه تأ ات جانبية side effects مثل إضافة حيوانات أخرى. لا أن النصف الثا لفريق التنفيذ،تستطيع بر ة هذه التأ ات ا انبية بإ ام النصف الثا للطريقة ا زئية.ولكن لو يكن هناك تأ ات جانبية ذا التنفيذ ا اص،فتستطيع فقط ترك الطريقة ا زئية غ منتهية.فهي اختيارية.

ولك أقول أنك إذا تكتب النصف الثا للطريقة ا زئية،فإن مترجم الفيجوال بيسك سيحذف كلا النصف ،توليد كود وكأن النصف الغ منفذ لن يتم توليده بشكل آ كان الأول.لذلك فئة ا يوان السابقة تصبح:

```
Partial Class Animal
    Public Sub Move()
        ' ----- Interesting movement code, then...
    End Sub
End Class
```

ليس فقط أن تعريف الطريقة ا زئية سيختفي،ولكن الاستدعاء الذي كان لتلك الطريقة `MoveSideEffects` داخل الروتMove سيختفي أيضاً.

مسائل متعلقة(قضايا مرتبطة با وضوع)Related Issues

دع أخذك لعدة دقائق هنا قبل الدخول كود ا شروع ناقشة بعض القضايا ال لا تتناسب مع مناقشة أي فصل خاص،ولكن من ا تمل اية ا طاق تستخدمها كث اً تطبيقاتك ا اصة.

الطريقة MsgBox

على الرغم من أنك قد استخدمتها تقريباً كل صفحة من هذا الكتاب ح الآن،فلم أقدم لك بشكل ر ي الطريقة "صندوق الرسالة MsgBox" جزء من فضاء ا اءMicrosoft.VisualBasic،"صندوق الرسالة MsgBox " تأتcarryover من الدالة MsgBox الإصدار الأصلي للفيجوال بيسك.فهي تعرض نافذة رسالة بسيطة، متضمنة اختيار ما لأزرار الاستجابة وأيقونة اختيارية.وكدالة،فهي تعود بكود يش إ أي زر نقره ا ستخدم لإغلاق الفورم،واحد من قيم العدادMsgBoxResult. التركيب هو التا :

```
Public Function MsgBox(ByVal Prompt As Object, Optional ByVal Buttons As MsgBoxStyle =
MsgBoxStyle.OkOnly,Optional ByVal Title As Object = Nothing) As MsgBoxResult
```

يقبل الوسيط Promptسلسلة حرفية stringلعرضها ا سم الرئيسي للحوار،تشButtons الأزرار أي زرbuttons، أيقونة icons وإعدادات أخرى للاستخدام عند عرض ا وار، والعنوان Titleيقبل عنوان نافذة صص إذا أردت أن يظهر شيء ما غ عنوان التطبيق.العبارة التالية تعرض النافذة التالية ا بينة الشكل التا :

```
Dim result As MsgBoxResult = MsgBox("It's safe to click; the computer won't explode.",
MsgBoxStyle.YesNoCancel Or MsgBoxStyle.Question, "Click Something")
```

Click Something

It's safe to click; the computer won't explode.

نعم | لا | إلغاء الأمر

الدالة MsgBox يتم اعتبارها لأن تكون جزء جوهري من اللغة.ولكن كجزء من فضاء أ اء Microsoft.VisualBasic،فهي تستخدم بشكل عام فقط ضمن لغة الفيجوال بيسك. ولكن لو أردت عمل شيء ما بكتابة كود ما السي شارب#C ، فلديك اختيار الطريقةMessageBox.Show. وهي ستعمل نسبياً مثل الدالةMsgBox، ولكن معاملا ا النسبية الثا والثالث تكون معكوسة.

استخدام Using DoEvents

يتم تصميم ال امج لعمل الكثـ من التفكـ ،وهي نسبياً تغلق الكمبيوتر.وهذا خاصةً صح،بالنسبة لطرق الفيجوال بيسك ال تعمل الكثـ من إجراءات قاعدة البيانات الثقيلة،واحد بعد الآخر.يؤجل defersالنظام ديثات الشاشة الأقل أ ية لذلك كن أن دث أولا كود معا ة البيانات الأكثر أ ية. وهذا عظيم،ولكن بعض الأحيان يفكر ا ستخدم فيقول"هذا الكمبيوتر الغ وت ثانية(تطول الاستجابة(مد))"ويسحب مقبس الكهرباء.إذا كانت الشاشة ستوفر ديثات أفضل،فإن ا ستخدم من ا تمل أن يصبح أكثر صـ patient .

كل أداة على وذجك(والفورم نفسها) تتضمن طريقة ديث(إنعاشRefresh)،ولكن من ا مكن أن يكون من ا تعب إنعاش كل شيء باستمرار.وإنعاش الشاشة(العرض)لن يعمل الكثـ لتمكـ زر"إلغاء" والذي تريد من مستخدمك أن ينقره لإجهاض كل معا ات (البيانات) ا ارية.تتضمن الفيجوال بيسك الطريقة DoEvents. عندما يتم استدعاء هذه الطريقة،فإن كود الطريقة ا الية يتوقف بشكل مؤقت،ويتم معا ة الرسائل مسار تنفيذ طابور الرسائل القادمة،متضمنة رسائل" paint "(ديث الشاشةscreen update).والطريقة DoEventsهي جزء من فضاء الأ اءMy،ويتم استخدامها كعبارة قائمة بذا ا.

```
My.Application.DoEvents()
```

كن حذر حيث أن الاستخدام ا فرط لـ DoEvents كن أن يبطئ تطبيقك،و كن أن يقود شاكل متعلقة بكون ا دث قد استدعاءه الكثـ من ا رات. بشكل عام ، ب أن يتم استخدامه مقطع كود معا ة مركز،ومن سيتم توسيعه ساحة أكـ يث يتم استدعاءه فقط عدة مرات كل ثانية على الأكثر.

ا عاملات النسبية (مصفوفة الوسيطاتParamArray) ParamArray Arguments

إن أي طريقة كن أن كن معاملات نسبيةoptional arguments،والكود ا ستدعي كن أن تار تضم أو إخراج هذه ا عاملات النسبية.ولكن ماذا إذا ما أردت إضافة عدد غ دد من ا عاملات النسبية الاختيارية لطريقة ما؟كيف كن أن تكتب هذا،على سبيل ا ثال،دالة ما،ستعود با توسط ميع ا عاملات ا زودة،بدون حدود على عدد ا عاملات النسبية؟و كن أيضاً أن تقبل متغ مصفوفة ما مع قيم الكود ا صدري،و كن أيضاً استخدام معامل نسـ مصفوفة الوسيطات*parameter array argument*، وتدعى أيضاً ا عامل النسـ ParamArray.وكما مع ا عاملات النسبية الاختيارية، ب أن تظهر ا عاملات النسبية عند اية قائمة ا عاملات النسبية لطريقة ما،و كن أن يكون معامل نسـ واحد فقط،ولأن واحد هو أكثر من كا بالنسبة لأي طريقة.تستخدم ا عاملات النسبية صفوفة ا عاملات الكلمة ا جوزة ParamArrayقبل اسم ا عامل النسـ اماً.

```
Public Function CalculateAverage(ByVal ParamArray sourceData() As Decimal) As Decimal
        ' ----- Calculate the average for a set of numbers.
        Dim singleValue As Decimal
        Dim runningTotal As Decimal = 0@
        If (sourceData.GetLength(0) = 0) Then
            Return 0@
        Else
            For Each singleValue In sourceData
                runningTotal += singleValue
            Next singleValue
            Return runningTotal / sourceData.GetLength(0)
        End If
    End Function
```

الاستدعاء للدالة CalculateAverageالآن يقبل أي عدد من القيم العشريةdecimal .

```
MsgBox(CalculateAverage(1, 2, 3, 4, 5)) ' Displays: 3
```

مشروع

يعا كود هذا الفصل ميزتـ شروع ا كتبة1☹فئة مساعدة بسيطة تستخدم مع أداة صندوق قائمة ListBoxوأداة صندوق مركب ComboBoxلإدارة النص والبيانات،و(2) موعة من النماذج الشاملة ا ستخدمة لتحرير جداول البحث ا كتبة مثل جداول أكواد ا الات.

دعم صناديق القائمة او ركب Supporting List and Combo Boxes

فيجوال بيسك6 والإصدارات الأقدم،كان صندوق القائمة ListBoxوالصندوق ا ركب ComboBoxيتضمنان مصفوفتان كالتجمعات:قائمة List (تستخدم لتخزين النص ا عروض لكل بند)وبيانات البندItemData (ا ستخدمة لتخزين قيمة عددية 32-بت لكل بند).كانت مصفوفة القائمة هامة بالنسبة للمستخدم فهي ال لب النص لكل بند.ولكن الكثـ من ا يعتمدون أكثر على مصفوفة بيانات البندItemData، وال تتيح معرف يز يتم إرفاقه مع كل بند قائمة.

```
cboMonth.AddItem("January")
        cboMonth.ItemData(cboMonth.NewIndex) = 1
        cboMonth.AddItem("February")
        cboMonth.ItemData(cboMonth.NewIndex) = 2
```

```
...
        cboMonth.AddItem("December")
        cboMonth.ItemData(cboMonth.NewIndex) = 12
```

فيما بعد،بعد أن تار ا ستخدم قيمة من القائمة، كن أن يتم استخدام ا عرف IDالرقمي للبحث   قاعدة البيانات أو لأهداف تصميمية أخرى.

```
nMonth = cboMonth.ItemData(cboMonth.ListIndex)
```

الأخبار السيئة أن كل من القائمة Listولا ح بيانات بند ItemDataموجودة   إصدار الدوت نت لصندوق القائمة ListBoxوالصندوق ا ركب ComboBox. ولكن الأخبار ا يدة هو أنه استبدال كل منهما تجميع أكثر مرونة بكثـ :البنودItems . ميع البنود زن أي نوع من الكائنات تريد:حالات انتغر،سلاسل حرفية،تاريخ،أي نوع أخر.وتستطيع مزجها ضمن صندوق قائمة مفرد. ا أن البنود   رد ميع فقط من نسخة(حالة) System.Object ، فتستطيع وضع أي نوع من الكائنات ال   ترغب   التجميعcollection . صندوق القائمة (أو الصندوق ا ركب)يستخدم هذا التجميع لعرض البنود القائمةlist .

بالتا   كيف يعرف صندوق القائمة ListBoxكيفية عرض النص لأي من مزيج من الكائنات؟بشكل افتراضي،تستدعي الأداة الطريقة "إ   نص ToString " للكائنات و ToStringمعرفه   System.Objectوتستطيع إعادة قياد ا   override فئتك ا اصة.تتضمن أداة صندوق القائمة ListBoxأيضاً ا اصية "عرض عضو DisplayMember " وال   تستطيع وضعهو قل أو خاصية فئتك ال تولد النص ا ناسب.

لنرى صندوق قائمة بالفعل.أضف صندوق قائمة ListBoxجديد للفورم،ومن أضف الكود التا   عا   حدث ميل الفورم.

```
Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        ListBox1.Items.Add(1)
        ListBox1.Items.Add("Easy")
        ListBox1.Items.Add(#5/3/2006#)
    End Sub
End Class
```

شغل التطبيق لاحظ أن الكود السابق يعرض الفورم كما   الشكل التا   .

Form1
1
Easy
05/04/1427

من أجل قيمة بيانات بند ItemDataالقد، تتضمن أداة صندوق القائمة ListBoxا اصية ValueMemberوال تعرف حقل ا عرف identifier أو خاصية من أجل الكائنات   ميع البنودItems . ولكن ليس عليك استخدام ValueMember.بدل عنها،تستطيع ببساطة استخراج الكائن حال الطلب من ميع البنودItems ، و ت أعضاءه من خلال كودك ا خصص لتحديد هويته. ا قيقة،هو عمل أكثر بقليل ا كان   طريقة الفيجوال بيسك6.ولكن مرة أخرى ا إنك تستطيع زين كائن بأي حجم   ميع البنودItems ، تستطيع اختيار زين كامل سجلات قاعدة بيانات،شيء ما   تكن تستطيع عمله قبل الدوت نت.

ولكن يبقى زين كامل السجلات   صندوق قائمة ListBoxأو صندوق مركب ComboBoxفيه الكثـ من الإسراف.والأكثر ملائمةً هو زين فقط عدد ا عرفID ، واستخدامه ككلمة ث   قاعدة البيانات.وهذا ما سنعمله   مشروع ا كتبة.لدعم هذا ، تاج إ إنشاء فئة بسيطة   يث تعرض قيمة النص والبيانات.أولاً،لنعود إ كود الكتبة.

لنضع الفئة   ملف الكود ا صدري   يث يصبح كله خاص   ا.أضف ملف فئة جديدة من خلال مشروع Project>> أمر القائمة أضف فئة Add Class.سم ا لف *ListItemData.vb* وانقر الزر إضافةAdd . سيظهر الكود التا   بشكل آ .

```
Public Class ListItemData

End Class
```

ستتضمن هذه الفئة فقط أعضاء لعرض بند ونص . حالة نسينا وصل حقل النص إ   ا اصية DisplayMemberلصندوق القائمة أو الصندوق ا ركب،سنضمن أيضاً إعادة قيادة للوظيفةToString ، زائد مشيد   صص والذي يعمل الإسناد الأ initializationللأعضاء بشكل أسهل.أضف الكود التا   سم الفئة:

```
Public Class ListItemData
    Public ItemText As String
    Public ItemData As Integer

    Public Sub New(ByVal displayText As String, ByVal itemID As Integer)
```

```
        ' ----- Initialize the record.
        ItemText = displayText
        ItemData = itemID
    End Sub

    Public Overrides Function ToString() As String
        ' ----- Display the basic item text.
        Return ItemText
    End Function

    Public Overrides Function Equals(ByVal obj As Object) As Boolean
        ' ----- Allow IndexOf() and Contains() searches by ItemData.
        If (TypeOf obj Is Integer) Then
            Return CBool(CInt(obj) = ItemData)
        Else
            Return MyBase.Equals(obj)
        End If
    End Function

End Class
```

فيما بعد عندما وقت تعبئة ListBox نستطيع استخدام هذا الكائن لإضافة وعرض قيم ا طابقة.

```
ListBox1.Items.Add(New ListItemData("Item Text", 25))
```

يتيح لنا إعادة قيادة الطريقة Equalsالبحث بسرعة بنود إضافتها سابقاً إ أداة صندوق القائمة(أو ما شابه)باستخدام ميزات مضمنة الأداة. ميع البنود Items لأداة صندوق القائمة يتضمن الطريقة IndexOfوال تعود وقع البند ا طابق.عادة،هذه الطريقة ستطابق الكائن نفسه فقط،إذا مررت ا حالة ListItemData،فإ ا ستعلن فيما إذا كان البند موجود سابقاً صندوق القائمةListBox. وكود Equalsا دث سيعود أيضاً بصواب Trueإذا مررت قيمة انتغر Integer value تتطابق مع عضو موجود القائمة سابقاً.

```
Dim itemPosition As Integer = SomeListBox.Items.IndexOf(5)
```

ديث جداول الكود Editing Code Tables

بالعودة إ الفصل الرابع،عندما عملنا على صناعة قاعدة البيانات شروع ا كتبة،فقد إنشاء العديد من ا داول لإملاء قوائم صندوق قائمة ComboBox بسيط التطبيق. يع هذه ا داول تكون مسبوقة بـ(Code) و توي سجلات نادراً ما تتغ دورة حياة التطبيق،كمثال ذه ا داول هو ا دول CodeCopyStatus،والذي دد الشرط(ا الة)العامة ا الية لبند معات collections ا كتبة.

Field	Type	Description
ID	Long - Auto	Primary key; automatically assigned. Required. مفتاح رئيسي،آلي الإسناد. مطلوب .
FullName	Text(50)	Name of this status entry. Required. اسم مدخلة الحالة.مطلوب .

ا أن يع هذه ا داول بشكل أساسي لديها نفس التنسيق – حقل ا عرف IDوواحد أو أكثر من ا قول ا تواة- فمن ا مكن تصميم قالب شامل يُستخدم لتحديث هذه ا داول.ستوفر فورم قاعدية(أو فئة) ميزات التحديث القاعدي،ليتم تطويرها بالكامل من خلال نسخ مشتقة derived versions من فئة النموذج القاعدي.

من أجل ا شروع،سنضيف وذج : وذج"ا لمخص summary"(والذي يعرض قائمة ميع الأكواد ا عرفة ا الية)و وذج"تفاصيلdetail "(والذي يسمح بتحديث كود موجود أو جديد مفرد). عل الأشياء ح أكثر بساطةً،سنضمن فقط التخصص الوظيفي لإدارة سجل قاعدي على الأكثر وذج"ا لمخص summary".معظم الكود الذي تاج لتحديث، عرض،وإزالة أكواد ستظهر اذج "التفاصيل detail ".

وذج التفاصيل الشامل The Generic Detail Form

أضف وذج جديد إ ا شروع (مشروع Project << وذج ويندوز جديدNew Windows Form)، ه BaseCodeForm.vb وغ ا صائص التالية كما هو مب :

اصية Property	الإعدادSetting
(Name)	BaseCodeForm
ControlBox	False
FormBorderStyle	FixedDialog
ShowInTaskbar	False
Size	406, 173
StartPosition	CenterScreen
Text	Code Form

والآن كن من الوصول إ الكود ا صدري ذه الفئة(عرض View << الكودCode).لن يعمل الكود على إنشاء حالات من هذه الفورم الشاملة مباشرةً،لذلك لنعمل على منع يع التناسخ instantiation ا باشر وذلك بتضم الكلمة ا جوزة MustInherit.

```
Public MustInherit Class BaseCodeForm
```

```
End Class
```

ا يزة الرئيسية للفورم سيكون إضافة سجلات كود جديدة،التحديث لسجلات كود موجودة،وإزالة سجلات موجودة.أضف ا يكل البنيوي لثلاث دوال والذي يدعم هذه ا يزات الثلاث.نستطيع أن عل هذه الوظائف MustOverride،ولكن كما سترى لاحقاً،سنحتاج يار حفظ التخصص الوظيفي الافتراضي من النموذج الشامل القاعدي.

```
 Public Overridable Function AddRecord() As Integer
        ' ----- Prompt to add a new record. Return the ID
        '       when added, or -1 if cancelled.
        Return -1
    End Function
    Public Overridable Function DeleteRecord(ByVal recordID As Integer) As Boolean
        ' ----- Prompt the user to delete a record.
        '       Return True on delete.
        Return False
    End Function
    Public Overridable Function EditRecord(ByVal recordID As Integer) As Integer
        ' ----- Prompt the user to edit the record. Return the
        '       record's ID if saved, or -1 on cancel.
        Return -1
    End Function
```

ستأخذ فئة "التفاصيل detail " مسؤولية ملء أداة صندوق القائمة ListBoxعلى فورم "الملخص summary" "بنودها. نحتاج لطريقتين من أجل معالجة هذا:واحدة تعمل على إضافة جميع البنود،وأخرى تحدِث بند وحيد.ستكون الفئة المشتقة ضرورية لدعم هذه الميزات.

```
' ----- Fill a ListBox control with existing records.
    Public MustOverride Sub FillListWithRecords( ByRef destList As ListBox, ByRef exceededMatches 
As Boolean)
    ' ----- Return the formatted name of a single record.
    Public MustOverride Function FormatRecordName(ByVal recordID As Integer) As String
```

يجب أن تعرض أيضاً فورم "التفاصيل" العنوان ومعلومات الاستخدام المناسبة على نموذج"الملخص".

```
    ' ----- Return a description of this editor.
    Public MustOverride Function GetEditDescription() As String
    ' ----- Return the title-bar text for this editor.
    Public MustOverride Function GetEditTitle() As String
```

على الرغم من أن معظم الجداول ستوفر قائمة قصيرة بالأكواد المرتبة أبجديًا،بعض الجداول ستتضمن عدد كبير(ربما الآلاف)من الأكواد.ستدعم فورم "الملخص summary " طريقة بحث،لإيجاد كود موجود بسرعة.بما أن النماذج المشتقة الخاصة ستستخدم هذه الميزة،فلن نضمن MustOverride.

```
Public Overridable Sub SearchForRecord(ByRef destList As ListBox, ByRef exceededMatches As Boolean)
        ' ----- Prompt the user to search for a record.
        Return
    End Sub
```

أخيراً،ستشير فورم التفاصيل إلى أيّاً من الميزات المتاحة التي يمكن استخدامها من فورم"الملخص".وستستدعي نموذج الملخص كل من الدوال التالية ومن ثم تمكِن أو لاتمكِن ميزات حسب الحاجة.

```
Public Overridable Function CanUserAdd() As Boolean
        ' ----- Check the security of current user to see
        '       if adding is allowed.
        Return False
    End Function
    Public Overridable Function CanUserEdit() As Boolean
        ' ----- Check the security of the user to see
        '       if editing is allowed.
        Return False
    End Function
    Public Overridable Function CanUserDelete() As Boolean
        ' ----- Check the security of the user to see
        '       if deleting is allowed.
        Return False
    End Function
    Public Overridable Function UsesSearch() As Boolean
        ' ----- Does this editor support searching?
        Return False
    End Function
```

هذا كل ما تحتاجه فورم التفاصيل الشاملةgeneric detail form. فيما بعد في هذا الكتاب،سنعمل على إنشاء إصدارات مشتقة لكل جدول كود code tables.

نموذج "الملخص" الشاملة.The Generic Summary Form

فورم ا لملخص أكثر بساطةً بقليل من النموذج السابق، ا ا أ ا عبارة عن فورم بسيط فقط.عندما تبدأ،فا ا تستخدم حالة(نسخة)لواحدة من اذج التفاصيل للتحكم بالاختبار(التجربة) ا لوب للمستخدم.أضف وذج جديد للفورم كما تعلمت من ما سبق و ه ListEditRecords.vb، عدل عليه بعض ا صائص يث يبدو كما هو مب الشكل التا .
لأداة صندوق قائمة ListBoxكب ة معظم الفورم،وهي الأداة ال ستحفظ يع البنود ا وجودة.يوجد أيضاً زر"إضافة" و" ديث"وزر "حذف"،وزر " ث" عن بنود القائمة.يوجد الكث من الكود لإدارة هذه البنود،وقد قمت بكتابتها سابقاً:(راجع مشروع هذا الفصل من أجل خاصيات وإعداد هذه الفورم).

انتقل إ الكود صدري للفورم،وأضف الكود ا ب فيما يلي:

```
Public Class ListEditRecordsvb
    Private DetailEditor As Library.BaseCodeForm

    Public Sub ManageRecords(ByRef UseDetail As Library.BaseCodeForm)
        ' ----- Set up the form for use with this code set.
        Dim exceededMatches As Boolean

        DetailEditor = UseDetail
        RecordsTitle.Text = DetailEditor.GetEditTitle()
        RecordsInfo.Text = DetailEditor.GetEditDescription()
        Me.Text = DetailEditor.GetEditTitle()
        ActAdd.Visible = DetailEditor.CanUserAdd()
        ActEdit.Visible = DetailEditor.CanUserEdit()
        ActDelete.Visible = DetailEditor.CanUserDelete()
        ActLookup.Visible = DetailEditor.UsesSearch()
        DetailEditor.FillListWithRecords(RecordsList, exceededMatches)
        RefreshItemCount(exceededMatches)
        Me.ShowDialog()
    End Sub
    Private Sub RefreshButtons()
        ' ----- Update the buttons as needed.
        ActEdit.Enabled = CBool(RecordsList.SelectedIndex <> -1)
        ActDelete.Enabled = CBool(RecordsList.SelectedIndex <> -1)
    End Sub
    Private Sub RefreshItemCount(ByVal exceededMatches As Boolean)
        If (exceededMatches) Then
            ResultCount.Text = "Results limited to first " & _
                RecordsList.Items.Count & " items"
        ElseIf (RecordsList.Items.Count = 1) Then
            ResultCount.Text = "1 record displayed"
        Else
            ResultCount.Text = RecordsList.Items.Count & " records displayed"
        End If
    End Sub
    Private Sub ActClose_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ActClose.Click
```

```
        ' ----- Close the form.
        DetailEditor.Close()
        DetailEditor = Nothing
        Me.Close()
    End Sub
    Private Sub ActAdd_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ActAdd.Click
        ' ----- Let the user add a record.
        Dim newID As Integer
        Dim newPosition As Integer

        ' ----- Prompt the user.
        newID = DetailEditor.AddRecord()
        If (newID = -1) Then Return

        ' ----- Add this record to the list.
        newPosition = RecordsList.Items.Add((New Library.ListItemData( _
            DetailEditor.FormatRecordName(newID), newID)))
        RecordsList.SelectedIndex = newPosition
        RefreshButtons()
        RefreshItemCount(False)
    End Sub
    Private Sub ActEdit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ActEdit.Click
        ' ----- Let the user edit a record.
        Dim newID As Integer
        Dim listItem As Library.ListItemData

        ' ----- Retrieve the item to edit.
        If (RecordsList.SelectedIndex = -1) Then Return
        listItem = CType(RecordsList.SelectedItem, Library.ListItemData)

        ' ----- Prompt the user.
        newID = DetailEditor.EditRecord(listItem.ItemData)
        If (newID = -1) Then Return

        ' ----- Modify the text of this item.
        listItem.ItemText = DetailEditor.FormatRecordName(newID)
        RecordsList.Items(RecordsList.SelectedIndex) = listItem
        RefreshItemCount(False)
    End Sub
    Private Sub ActDelete_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ActDelete.Click
        ' ----- Let the user edit a record.
        Dim listItem As Library.ListItemData

        ' ----- Retrieve the item to edit.
        If (RecordsList.SelectedIndex = -1) Then Return
        listItem = CType(RecordsList.SelectedItem, Library.ListItemData)

        ' ----- Prompt the user.
        If (DetailEditor.DeleteRecord(listItem.ItemData) = False) Then Return

        ' ----- Remove this item from the display.
        RecordsList.Items.RemoveAt(RecordsList.SelectedIndex)
        RefreshButtons()
        RefreshItemCount(False)
    End Sub
    Private Sub ActLookup_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ActLookup.Click
        ' ----- Prompt for user limitation of the displayed items.
        Dim exceededMatches As Boolean

        DetailEditor.SearchForRecord(RecordsList, exceededMatches)
        RefreshButtons()
        RefreshItemCount(exceededMatches)
    End Sub
    Private Sub RecordsList_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
RecordsList.Click
```

```
        ' ----- Enable controls as needed.
        RefreshButtons()
    End Sub
    Private Sub RecordsList_DoubleClick(ByVal sender As Object, ByVal e As System.EventArgs)
Handles RecordsList.DoubleClick
        ' ----- Same as the edit button.
        If (ActEdit.Visible = True) Then ActEdit.PerformClick()
    End Sub
    Private Sub RecordsList_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles RecordsList.SelectedIndexChanged
        ' ----- Enable controls as needed.
        RefreshButtons()
    End Sub
End Class
```

السطر الأول للكود ا ضاف يعرف حالة خاصة من وذج"التفاصيل"الشاملة وال صممناها منذ قليل(الفورم السابقة ذه الفورم).

```
Private DetailEditor As Library.BaseCodeForm
```

هذا ا قل فظ حالة فئة مشتقة من الفورم BaseCodeForm.وذلك الإسناد يظهر طريقة عامة ManageRecords .

```
Public Sub ManageRecords(ByRef UseDetail As Library.BaseCodeForm)
```

أرجع إ كود هذه الطريقة،ترى أن الكود الذي يستدعي ManageRecords رر حالة فئة، واحدة من اذج BaseCodeForm ا شتقة.حا ا يتم الإسناد إ حقل داخلي DetailEditor، يستخدم الكود ا يزات العامة لتلك ا الة لإعداد العنصر ا عروض على وذج"ا لمخص" summary."على سبيل ا ثال،دالة وذج "التفاصيل " detail CanUserAdd وال تدعم قيمة معادة من نوع منطقي Boolean، تضع خاصية "مرئي " Visible لزر "إضافة ActAdd ".تستدعي الطريقة FillListWithRecords تعبئة أداة وذج"ا لمخص" ListBox بأي قيمة كود موجودة.بعد عمل بعض التعديلات ا عروضة،تعرض الطريقة Me.ShowDialog فورم"ا لمخص"للمستخدم.

على الرغم من أن ا ستخدم سيتفاعل مع الأدوات على وذج "ا لمخص"،فإن معظم هذه الأدوات ترجئ معا تها إ وذج"التفاصيل" أي DetailEditor. على سبيل ا ثال،نقر الزر"إضافة"يوجل معظم ا نطق إ طريقة وذج التفاصيل AddRecord.الكود وذج"ا لمخص"لا يعمل أكثر من ديث حقوله ا عروضة ا اصة.

```
Private Sub ActAdd_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ActAdd.Click
        ' ----- Let the user add a record.
        Dim newID As Integer
        Dim newPosition As Integer

        ' ----- Prompt the user.
        newID = DetailEditor.AddRecord()
        If (newID = -1) Then Return

        ' ----- Add this record to the list.
        newPosition = RecordsList.Items.Add((New Library.ListItemData( _
            DetailEditor.FormatRecordName(newID), newID)))
        RecordsList.SelectedIndex = newPosition
        RefreshButtons()
        RefreshItemCount(False)
    End Sub
```

معظم الكود ا تبقي وذج "ا لمخص" إما مشابه ذا(من أجل ميزات التحديث وا ذف والبحث)،أو يتم استخدامه لتنشيط العرض ا عتمد على تفاعل ا ستخدم مع الفورم.كن متأكد من اختبار الكود لتحصل على فهم جيد لكيفية عمل الكود. الفصول اللاحقة،عندما نضيف اذج التفاصيل العملية،سنرى هذا الكود بشكل عملي.

البرمجة الوظيفية Functional Programming

في هذا الفصل سنغطي موضوعي برمجة رئيسين في الفيجوال بيسك:تعابير الوسيط (لامداLambda)ومعالجة الأخطاء*error handling* .وكلاهما غامض،الأول بسبب استخدامه الحرف الإغريقي في اسمه والأخر لأنه بالإضافة لكونه بالإغريقي فهو صعب بالنسبة لكل المبرمجين.يجب أن يتم عمل تعابير الوسيط لامدا بمبدأ شامل للبرمجة الوظيفية*functional programming*، الفكرة هي أن كل مهمة معالجة يمكن أن يتم التعبير عنها كدالة(وظيفة)وتلك الدالة يمكن أن يتم تمريرها طوعاً أو كرهاً willy-nilly ضمن الكود المصدري.

الفيجوال بيسك ليست لغة برمجة وظيفية (برمجة بالدوال) حقيقية،ولكن تقديم تعابير الوسيط لامدا lambdaفي الفيجوال بيسك2008 يجلب للغة بعض هذه الطرق الوظيفية والوسائل.

تعابير الوسيط لامدا Lambda Expressions

يتم تسمية تعابير الوسيط لامدا لحسابات لامدا*lambda calculus* (أو λ-calculus)،نظام رياضي تم تصميمه في 1930sبواسطة ألونزو كارك Alonzo Church،على الرغم من أن عمله كان في البداية نظرياً،فإنه قاد إلى ميزات وتراكيب تفيد معظم لغات البرمجة اليوم.على وجه التخصيص،توفر حسابات لامدا الأساس المنطقي rationale لدوال الفيجوال بيسكfunctions، والمعاملاتarguments، والقيم المعادة return values التي درسناها سابقاً.لذلك ما الفائدة من إضافة ميزة جديدة للفيجوال بسك وتسميتها lambda بما أن أشياء الوسيط لامدا موجودة في اللغة؟سؤال عظيم ،ولكن لاجواب.

تتيح لك تعابير لامدا من تعريف كائن ما يحتوي على دالة كاملة.على الرغم من أن هذا نوعاً ما شيء جديد في الفيجوال بيسك.فميزة مشابهة قد كانت موجودة في لغة البيسك BASICولفترة طويلة.وقد وجدت صيغة يدوية قديمة من لغة البرمجة الأقدم التي استخدمتهاBASIC PLUS، فكانت توفر العبارة DEF،التي تتيح لك تعريف دوال بسيطة.إليك بعض مثال الكود من تلك والتي تطبع قائمة من خمس تربيعات

```
100:    DEF(SQR(X)  = X * X)
110:    For I = 1 To 5
120:        Print(I, SQR(I))
130:    Next I
140:    End
```

يظهر تعريف الوظيفة(الدالة)في السطر100،تعود بمربع أي معامل نسبي ممرر لها.وتستخدمه في النصف الثاني في السطر120،مولدة المخرجات التالية:

1 1
2 4
3 9
4 16
5 25

تعمل تعبيرات لامدا في الفيجوال بيسك بطريقة مشابهة،تتيح لك تعريف متغير كدالة بسيطة.إليك مكافئ الفيجوال بيسك من أجل الكود السابق:

```
Dim sqr As Func(Of Integer, Integer) = Function(x As Integer) x * x
        For i As Integer = 1 To 5
MsgBox(i & vbTab & sqr(i))
        Next i
```

تعبير لامدا الفعلي في السطر الثاني:

```
Function(x As Integer) x * x
```

تبدأ تعابير لامدا بالكلمة المحجوزةFunction،متبوعة بقائمة المعاملات النسبية الممررة داخلها passed-in ضمن أقواس.بعد ذلك يأتي تعريف الدالة نفسها،يستخدم تعبير ما `x * x` المعاملات الممررة داخلياً لتوليد نتيجة ما نهائية.في هذه الحالة النتيجة هي قيمة xمضروبة بنفسها.

شيء ما لن تراه في تعبير لامدا وهو عبارة Return.عوضاً عنه، تبدو القيمة الراجعة واقعة خارج نطاق التعبير بطبيعة الحال.وهذا لما تحتاج نوع ما من متغير لحفظ التعريف definitionوالراجع returnوالنتيجة resultفي تركيب مشابه لدالة.

```
Dim sqr As Func(Of Integer, Integer)
```

يتم تعريف متغيرات تعبير لامدا باستخدام الكلمة المحجوزةFunc. تطابق قائمة نوع بيانات المعامل النسبي قائمة المعاملات النسبية لتعبير لامدا الفعلي،ولكن مع نوع بيانات إضافي ملحق في النهاية `Integer` والذي يمثل نوع بيانات القيمة العائدة.إليك تعبير لامدا الذي يتفحص فيما إذا يعمل معامل نسبي عددي صحيح Integer على إرجاع أو عدم إرجاع نتيجة منطقيةBoolean .

```
Public Sub TestNumber()
        Dim IsEven As Func(Of Integer, Boolean) = Function(x As Integer) (x Mod 2) = 0
        MsgBox("Is 5 Even? " & IsEven(5))
    End Sub
```

يعرض هذا الكود رسالة تقول"هل يقبل5القسمة على 2؟خطأ"خلف المشهد،تولد الفيجوال بيسك دالة فعلية،وتربطها بالمتغير الذي يستخدم التفويض(التفويض كما تتذكر، هو أسلوب لتعين طريقة ما وبشكل عام من خلال متغير متمايز) الكود التالي هو حقيقة ما يولده المترجم من أجل عينة الكود السابق.

```
Private Function HiddenFunction1(ByVal x As Integer) As Boolean
        Return (x Mod 2) = 0
    End Function
    Private Delegate Function HiddenDelegate1(ByVal x As Integer) As Boolean
    Public Sub TestNumber()
        Dim IsEven As HiddenDelegate1 = AddressOf HiddenFunction1
        MsgBox("Is 5 Even? " & IsEven(5))
    End Sub
```

في هذا الكود،تعبير لامدا والمتغير IsEvenالمتعلق به قد تم تبديلهما بدالة حقيقية(HiddenFunction1)وتفويض وسيط(HiddenDelegate1).على الرغم من أن الوسائط لامدا جديدة في الفيجوال بيسك2008،فإن هذا النوع من التخصص الوظيفي المكافئ قد أصبح ممكناً منذ نسخة الفيجوال بيسك الأولى للدوت نت.توفر تعابير لامدا أبسط تركيب عندما تعمل الدالة المرجعية التفويض delegate-referenced function على إرجاع النتيجة من التعبير فقط.

لقد تم إضافة تعابير لامدا للفيجوال بيسك2008 بشكل رئيسي لدعم التخصص الوظيفي الجديد لينكو(الفصل17).وهي مفيدة بشكل خاص عندما تحتاج لتوفير تعبير ما كقاعدة(قانون) معالجة كود أخر،وخاصة الكود المكتوب بواسطة المشارك الثالث(عامل أجنبي).وفي تطبيقاتنا الخاصة،المشارك الثالث(العامل الأجنبي) هو ميكروسوفت.

تضمين وسيطات لامدا Implying Lambdas

تعابير لامدا جيدة ،ولكن من الواضح أن تخصص وظيفي مكافئ كان موجود سابقاً في اللغة.وبنفسها تعابير لامدا هي فقط تبسيط لتركيب تفويض- دالة -function delegateغير مرتب(فوضوي).ولكن عندما تضم تعابير لامدا مع ميزات استنتاج النوع *type inference* فسوف تحصل على شيء ما أفضل.ما تحصل عليه بالأنواع المستنتجة هو تعابير لامدا.وهذا ليس اسماً رومانسياً،ولكنه أداة عظيمة جديدة.لنقول أنك أردت كتابة تعبير لامدا يعمل على ضرب عددين ببعضهما.

```
Dim mult As Func(Of Integer, Integer, Integer) = Function(x As Integer, y As Integer) x * y
        MsgBox(mult(5, 6)) ' Displays 30
```

إن هذا نقل لقالب كبير من الكود:إني أخبر الفيجوال بيسك كل شيء ،وهي تطيعني بدون تردد. ولكن توجد أيضاً ترجمة أكثر حرية laissez faire للكود الذي يجلب استنتاج النوع ضمن التشغيل.

```
Dim mult = Function(x As Integer, y As Integer) x * y
```

هذا كود أقل بكثير.لقد تعبت من كتابة *Integer* مرة بعد مرة.يعمل الكود لأن الفيجوال بيسك نظر إلى ما هو مسند لـ multوبشكل مناسب عين(حدد)نوع بياناته القوية.وفي هذه الحالة، mult هو من نوع Function(Integer, Integer) As Integer (شاهد الشكل التالي).وحتى أنه وبشكل مناسب خمن النوع الراجع(المعاد).

```
Dim mult = Function(x As Integer, y As Integer) x * y
        Dim mult As <Function(Integer, Integer) As Integer>
```

يفترض أنك قد وضعت "خيار الاستنتاجOption Infe"ل"فعال On " في كودك المصدري،أو من خلال صفحات خاصيات المشروعProject properties (وهو الافتراضي).نستطيع اختصار تعريف multحتى لأكثر من ذلك.

```
Dim mult = Function(x, y) x * y
```

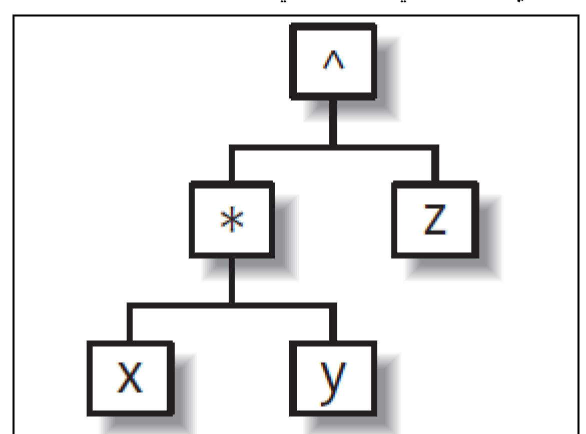
في هذا السطر سيستنتج الفيجوال بيسك نفس الدالة،ولكنه سيستخدم نوع بيانات الكائن Object على طول محل العددي الصحيحInteger. أيضاً،إذا كان لديك "خيار التدقيق Option Strict "وقد تم وضعه إلى"فعالOn" (والذي يجب عليك)،فهذا السطر لن يتم ترجمته حتى تضيف شرط As المناسب.

أشجار التعبير Expression Trees

داخلياً،يعمل الفيجوال بيسك على تغيير تعبير لامدا إلى "شجرة تعبير"أي تركيب هرمي يجعل العوامل operands مصاحبة لمعاملاتهاoperators. خذ هذا التعبير الشبه معقد الذي يرفع تعبير مضروب إلى قوة.

```
Dim calculateIt = Function(x, y, z) (x * y) ^ z
```

يولد الفيجوال بيسك شجرة تعبير للمتغير calculateItوالذي يبدو كما في الشكل التالي.



عندما يحين وقت استخدام تعبير لامدا،يدور الفيجوال بيسك على الشجرة،يحسب القيم من المستويات الأدنى إلى الأعلى.ويتم تخزين شجرات التعبير هذه ككائنات معتمدة على فئات في فضاء أسماء System.Linq.Expressions.إذا كنت لا تحب كتابة تعابير لامدا،بإمكانك بناء شجرات تعبيرك الخاصة باستخدام هذه الكائنات. ومهما يكن،لن أخوض بهذا الموضوع.

الوسيطات لامدا المعقدة Complex Lambdas

على الرغم من أن تعابير لامدا لايمكن أن تحتوي عبارات فيجوال بيسك مثل الحلقات For...Next،مايزال بإمكانك بناء بعض الحسابات المعقدة نوعاً ما باستخدام معاملات قياسية. الاستدعاءات الخارجية للدوال يمكن أيضاً أن تظهر في الوسيطات لامدا.في عينة الكود التالي يؤجل mult عمله للدالة MultiplyIt .

```
Private Sub DoSomeMultiplication()
        Dim mult = Function(x As Integer, y As Integer) MultiplyIt(x, y) + 10
        MsgBox(mult(5, 6)) ' Displays 40
    End Sub
    Public Function MultiplyIt(ByVal x As Integer, ByVal y As Integer) As Integer
        Return x * y
    End Function
```

هذا بسيط جداً.ولكن تحصل أشياء أكثر أهمية عندما يكون لديك تعابير لامدا تعود بتعابير لامدا أخرى.تم اختراع حسابات لامدا بشكل جزئي لرؤية كيف يمكن أن يتم تقسيم دالة معقدة في عدة دوال أكثر بساطة.وحتى القيم الحرفية(المحرفية) يمكن أن يتم تعريفها كوسيطات لامدا.إليك تعبير لامدا يعود دائما بالقيمة3.

```
Dim three = Function() 3
```

لقد رأيت منذ حين تعابير لامدا التي تقبل أكثر من معامل نسبي:

```
Dim mult1 = Function(x As Integer, y As Integer) x * y
```

في حسابات لامدا،يمكن أن يتم تقسيم هذا إلى دوال ابسط،حيث كل منها يتضمن فقط معامل نسبي وحيد.

```
Dim mult2 = Function(x As Integer) Function(y As Integer) x * y
```

نوع بيانات mult2ليس نفس نوع بيانات mult1، ولكن كلاهما يولد نفس الجواب من نفس قيم xو y. عندما تستخدمmult1، فإنه يحسب الناتج ل xو yويعيد به.وعندما تستخدمmult2، فإنه يشغل أولاً الجزء Function(x As Integer)،والذي يعود بـ لامدا أخر محسوب بتمرير قيمة xفي تعريفه.إذا مررت فيه "5" كقيمة لـx، فإن لامدا المعادة هي: `Function(y As Integer) 5 * y`

و لامدا هذا يتم حسابه،والناتج من 5و yيتم إعادته.واستدعاء mult2في الكود أيضاً مختلف قليلاً.فلا تمرر في المعاملين النسبيين في نفس الوقت.بدل ذلك،تمرر في المعامل النسبي لأجل x،ومن ثم تمرر لـy لـ لامدا الأولي المعاد به.

```
MsgBox(mult2(5)(6))
```

عندما تشغيل الجزء(5)mult2، يتم تبديله بـ لامدا المعاد أولاً.ومن ثم لامدا المعاد أولاً يتم معالجته باستخدام(6) كمعامله النسبيy. أليس هذا بالبسيط؟.

الجزء الهام الذي يجب تذكره هو أنه من الممكن بناء تعابير لامدا معقدة صعوداً من تعابير لامدا الأساسية(القاعدية).ستستخدم الفيجوال بيسك هذه الحقيقة عند توليد كود من أجل التعابير المتعلقة بلينكوLINQ. وسنتحدث عنها في الفصل 17،وحتى ذلك الحين،ستدير الفيجوال بيسك الكثير من تعابير لمدا التي تركز على لينكو LINQ من أجلك خلف المشهد(غير مرئية).

رفع (ترقية) المتغير Variable Lifting

على الرغم من أنك تستطيع تمرير معاملات نسبية ضمن تعبير لمدا،من المحتمل أيضاً أن تستخدم متغيرات أخرى ضمن مجال تعبير لمدا.

```
Private Sub NameMyChild()
        Dim nameLogic = GetChildNamingLogic()
        MsgBox(nameLogic("John")) ' Displays: Johnson
    End Sub
    Private Function GetChildNamingLogic() As Func(Of String, String)
        Dim nameSuffix As String = "son"
        Dim newLogic = Function(baseName As String) baseName & nameSuffix
        Return newLogic
    End Function
```

تعود الدالة GetChildNamingLogicبتعبير لمدا.وتعبير لمدا ذاك يتم استخدامه في الطريقةNameMyChild بتمرير Johnكمعامل نسبي لـ لامدا.وهو يعمل،ولكن السؤال كيف.المشكلة هي nameSuffix،المستخدم في منطق تعبير لمدا،وهو متغير محلي ضمن الطريقةGetChildNamingLogic. جميع المتغيرات المحلية يتم تدميرها عندما تخرج الطريقة.في الوقت الذي يتم فيه استدعاء الدالةMsgBox، سيكون nameSuffixقد انتهى منذ وقت.ولكن مايزال الكود يعمل وكأن nameSuffix مستمر. لجعل هذا الكود يعمل، تستخدم الفيجوال بيسك ميزة جديدة تدعى رفع المتغير*variable lifting*. رؤية nameSuffixسيتم الوصول له من خارج نطاق(مدى) GetChildNamingLogic.تعيد الفيجوال بيسك كتابة كودك المصدري،تعدل متغير محلي إلى متغير له مدى أوسعwider scope .

في النسخة الجديدة من الكود المصدري،تضيف الفيجوال بيسك فئة الإغلاق(النهاية)*closure class*):الفئة المولدة بشكل ديناميكي والتي تحتوي على كل من تعبير لامدا والمتغيرات المحلية المستخدمة بواسطة التعبير.عندما تضمهما مع بعضهما البعض،فأي كود يتمكن من الوصول لتعبير لمدا ستكون له أيضاً إمكانية الوصول للمتغير المحلي.

```
Private Sub NameMyChild()
        Dim nameLogic = GetChildNamingLogic()
        MsgBox(nameLogic("John")) ' Displays: Johnson
    End Sub
    Public Class GeneratedClosureClass
        Public nameSuffix As String = "son"
        Public newLogic As Func(Of String, String) = Function(baseName As String) baseName & Me.nameSuffix
    End Class
    Private Function GetChildNamingLogic() As Func(Of String, String)
        Dim localClosure As New GeneratedClosureClass
        localClosure.nameSuffix = "son"
        Return localClosure.newLogic
    End Function
```

الكود الحقيقي المولد بواسطة الفيجوال بيسك هو أكثر تعقيداً من هذا،وسيتضمن كل كود التحويل تفويض-دالة الذي كتبت حوله سابقاً.ولكن هذه هي الفكرة الأساسية.
فئات الإغلاق ورفع (ترقية)المتغير هي ميزات أساسية لتعابير لمدا بما أنك لا تعرف أين مكان تعابير لامدا على الإطلاق.

بوادئ الكائنات Object Initializers

للبدء،إن خاصيات الكائن object properties غير مدارة بواسطة المشيدات،تحتاج لإسناد هذه الخاصيات بشكل منفصل بعد إنشاء حالة فئة مباشرة.

```
Dim newHire As New Employee
        newHire.Name = "John Doe"
        newHire.HireDate = #2/27/2008#
        newHire.Salary = 50000@
```

توفر العبارة Withتركيب أفضل نوعاً ما.

```
Dim newHire As New Employee
        With newHire
            .Name = "John Doe"
            .HireDate = #2/27/2008#
```

```
            .Salary = 50000@
        End With
```

تم في الفيجوال بيسك2008 تضمين بناء syntaxجديد يتيح لك دمج التصريح(بالكلمة المحجوزةNew)وإسناد الأعضاء.يتضمن البناء نوع جديد لعبارة With.

```
Dim newHire As New Employee With {.Name = "John Doe", .HireDate = #2/27/2008#, .Salary = 50000@}
```

معالجة الأخطاء في الفيجوال بيسك Error Handling in Visual Basic

تصحيح ومعالجة الأخطاء هما اثنان من معظم نشاطات البرمجة الأساسية التي تقوم بها في أي وقت.حتى في التطبيق الخالي التصحيح،يوجد دائماً سبب للاعتقاد بأن المستخدم سيفقد أشياء غاية في الأهمية،عملك أن تكون حارس على بيانات المستخدم عند إدارتها بواسطة المستخدم،ولحفظها آمنة،حتى من تهاون المستخدم نفسه، وحتى من قبل كودك المصدري نفسه.إن أكثر من 50% من الكود يكون محجوز لمعالجة الأخطاء،البيانات السيئة،استثناءات النظام،والإخفاقات،بشكل مركزي،كل هذا الكود الإضافي يبطئ التطبيق ويضيف الكثير من التكاليف غير المباشرة لما تم تسميته "البرمجيات المنتفخةbloatware ".

جوهر الأخطاء في الفيجوال بيسك The Nature of Errors in Visual Basic

سوف تتعامل مع ثلاث تصنيفات رئيسية للأخطاء في تطبيقات الفيجوال بيسك:

أخطاء وقت الترجمة *Compile-time errors*

بعض الأخطاء هي واضحة (صريحة)جداً بحيث أن الفيجوال بيسك سوف ترفض ترجمة تطبيقك.بشكل عام،بعض الأخطاء تنتمي لقضايا بنائية بسيطة والتي يمكن أن يتم تصحيحها بعدة ضربات مفاتيح.ولكن تستطيع أيضاً تمكين ميزات في برنامجك التي تزيد عدد الأخطاء المميزة من قبل المترجم.على سبيل المثال،إذا وضعت خيار التدقيق Option Strict لـ"الفعال On " في تطبيقك أو ملف الكود المصدري،فإن تحويلات التضييق ستولد أخطاء وقت الترجمة.

```
' ----- افرض: Option Strict On
        Dim bigData As Long = 5&
        Dim smallData As Integer
        ' ----- The next line will not compile.
        smallData = bigData
```

تتضمن الفيجوال أستوديو2008 الميزات التي تساعدك في إيجاد وحل أخطاء وقت الترجمة.مثل هذه الأخطاء تم تعليمها بـ"بخط أزرق متعرجblue squiggle " بعض الأخطاء تطلب أيضاً من الفيجوال بيسك عرض خيارات تصحيح من خلال نافذة الانسدال(السياق)كما هو مبين في الشكل.

```
1  Option Strict On
2  Public Class Form1
3      Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
4
5          Dim bigData As Long = 5&
6          Dim smallData As Integer
7          ' ----- The next line will not compile.
8          smallData = bigData
9      End Sub
10 End Class
11
```

```
Option Strict On disallows implicit conversions from
'Long' to 'Integer'.

Replace 'bigData' with 'CInt(bigData)'.

        Dim smallData As Integer
        ' ----- The next line will not compile
        smallData = CInt(bigData)bigData
    End Sub
End Class

Expand All Previews
```

أخطاء وقت التنفيذ *Runtime errors*

تحدث أخطاء وقت التنفيذ عندما يسبب جمع(تركيب) من بيانات وكود شرط غير صحيح فيما يظهر من كود على أنه صحيح(محقق).مثل هذه الأخطاء تحدث باستمرار عندما يدخل المستخدم بيانات غير صحيحة في التطبيق،ولكن كودك الخاص يمكن أن يولد أخطاء وقت التنفيذ.خذ المقطع التالي من الكود:

```
Public Function GetNumber() As Integer
        ' ----- Prompt the user for a number.
        ' Return zero if the user clicks Cancel.
        Dim useAmount As String
        ' ----- InputBox returns a string with whatever
        ' the user types in.
        useAmount = InputBox("Enter number.")
        If (IsNumeric(useAmount) = True) Then
            ' ----- Convert to an integer and return it.
            Return CInt(useAmount)
        Else
            ' ----- Invalid data. Return zero.
            Return 0
        End If
    End Function
```

يبدو هذا الكود معقول لحد ما،وفي معظم الحالات،هو كذلك.فهو يطلب من المستخدم عدد ما،ويحول أعداد صحيحة لتنسيق عددي صحيح،ويعود بالنتيجة.الدالة IsNumericتجتث (تستأصل)أي مدخلات غير عددية غير صحيحة.استدعاء لهذه الدالة سيعود بالحقيقة بأعداد صحيحة محققة للقيم العددية المدخلة،و0 للمدخلات غير الصحيحة.

ولكن ماذا إذا ما تم إدخال قيمة مثل"342304923940234"وبما أنها قيمة عددية محققة،فسيتم تمريرها للدالة IsNumericولكن بما أنها تتجاوز حجم نوع بيانات العددي الصحيح،فإنها ستولد خطأ وقت التنفيذ كما هو مبين في الشكل التالي.



بدون كود إضافي لمعالجة الأخطاء أو يتفحص حدود البيانات المحققة(الصحيحة)،فإن الروتين GetNumberسيولد خطأ وقت التنفيذ هذا،ومن ثم يؤدي إلى إجهاض كامل البرنامج.

الأخطاء المنطقية *Logic errors*

الأخطاء المنطقية هي الثالثة،وهي أكثر أنواع الأخطاء انتشاراً.المتسبب بها هو أنت،المبرمج،لاتستطيع لوم المستخدم على هذه الأخطاء.من قضايا سياق المعالجة إلى الحسابات غير الصحيحة،الأخطاء المنطقية هي لوح تطوير البرمجيات وللحصول على نتيجة تحتاج إلى وقت أكثر للتصحيح من كلا النوعين السابقين مع بعضهما. تم تخصيص الكثير للأخطاء المنطقية في هذا الكتاب.تستطيع أن تجبر العديد من الأخطاء المنطقية خارج نطاق كودك بإضافة اختبارات كافية للبيانات الغير صحيحة،وبشكل مناسب اختبار تطبيقك تحت شروط وظروف متنوعة.

لن يكون لديك الكثير من الصعوبة في التعامل مع أخطاء وقت الترجمةcompile-time errors .فهم عام لمبادئ البرمجة في الدوت نت والفيجوال بيسك،واستخدام نظامي للأدوات المضمنة مع الفيجوال أستوديو2008،سيساعدك في سرعة إيجاد والتخلص من هذه الأخطاءcompile-time errors .

القضية الأكبر هي:ماذا تعمل بخصوص أخطاء وقت التنفيذ؟حتى ولو تفحصت كل البيانات الممكنة وشروط المصادر الخارجية،من المحتمل أن تمنع جميع أخطاء وقت التنفيذ.لن تعلم أبداً متى ستفشل عملية الاتصال بالانترنت،أو يخطئ المستخدم في كبل الطابعة،أو خدش على ديفيدي DVDسيعطي بيانات فاسدة. في أي وقت تتعامل مع المصادر الموجودة خارج كودك المصدري،فستكون لديك فرصة أكبر لحدوث أخطاء وقت التنفيذ.

يبين الشكل السابق ما تعمل الفيجوال بيسك عندما تصادف خطأ وقت التنفيذ:فهي تعرض للمستخدم حوار خطأ شامل،وتوفر فرصة لتجاهل الخطأ(من المحتمل فساد أي من البيانات الغير محفوظة)أو الخروج من البرنامج مباشرةً(فقدان كامل للبيانات الغير محفوظة).

على الرغم من أن هذين الفعليين يتركان للمستخدم الكثير من السحر،ولكن لا يعطيان طابع الثقة للمستهلك في مهارتك بكتابة الكود.إن المستخدم سيلومك لأي أخطاء تنتج بواسطة تطبيقك،وحتى لو كانت المشكلة الحقيقية قد تم إزالتها تماماً من كودك.

لحسن الحظ،تتضمن الفيجوال بيسك ثلاث أدوات لمساعدتك في التعامل بالكامل مع أخطاء وقت التنفيذ،إذا حدثت وعندما تحدث هذه الأخطاء.هذه الميزات الثلاث للفيجوال بيسك :معالجة الأخطاء غير التركيبيةunstructured error handling، معالجة الأخطاء التركيبيةstructured error handling، ومعالجة الأخطاء الغير معالجةunhandled error handling. يمكن أن يتم استخدام كل من هذه الثلاث في تطبيق الفيجوال بيسك لمنع بيانات المستخدم-والمستخدم نفسه- من أخطاء غير مراده.

معالجة الأخطاء الغير تركيبية Unstructured Error Handling

لقد أصبحت معالجة الأخطاء الغير تركيبية جزء من الفيجوال البيسك منذ الظهور الأول في عام1990.وهو سهل الاستخدام،يلتقط كل الأخطاء الممكنة في مقطع من كود،ويمكن تمكينه أو تعطيله حسب الحاجة.بشكل افتراضي،لاتتضمن إجراءات الطرق والخصائص أي معالجة للأخطاء على الإطلاق،لذلك عليك إضافة كود معالج الخطأ-التركيبي والغير تركيبي لكل إجراء حيثما تحس أن هناك حاجة له.

الفكرة التي تقع خلف معالجة الأخطاء الغير تركيبية هي إلى حدما أساسية.ببساطة تضيف سطر في كودك يقول"إذا حدث أي خطأ بأي حال،اقفز بشكل مؤقت إلى هذا المقطع الآخر من إجرائي حيث لدي كود خاص للتعامل معه.هذا"المقطع الآخر" يدعى"معالج الخطأ *error handler* ."

```
Public Sub ErrorProneRoutine()
        ' ----- Any code you put here before enabling the
        ' error handler should be pretty resistant to
        ' runtime errors.
        ' ----- Turn on the error handler.
        On Error GoTo ErrorHandler
        ' ----- More code here with the risk of runtime errors.
        ' When all logic is complete, exit the routine.
        Return
ErrorHandler:
        ' ----- When an error occurs, the code temporarily jumps
        ' down here, where you can deal with it. When you're
        ' finished, call this statement:
        Resume
        ' ----- which will jump back to the code that caused
        ' the error. The "Resume" statement has a few
        ' variations available. If you don't want to go
        ' back to main code, but just want to get out of
        ' this routine as quickly as possible, call:
        Return
    End Sub
```

تمكن عبارة On Error أو تعطل معالجة الخطأ في الإجراء.عندما يحدث خطأ ما،تضع الفيجوال بيسك تفاصيل ذلك الخطأ في كائن خطأ شامل global Err object .
وهذا الكائن يخزن نص وصفي للخطأ(الكود العددي للخطأ (إذا كان ممكناً))أو تفاصيل مساعدة عبر الانترنت فيما يتعلق بالموضوع،وقيم أخطاء نوعية.وسأجدول التفاصيل بعد قليل.
تستطيع تضمين العديد من عبارات On Error في كودك كما تريد.وكل واحدة يمكن أن توجه كود منحرف إلى رقعة (عنوان)مختلف.يمكن أن يكون لديك معالج خطأ وحيد لأخطاء الشبكة،وواحد خاص بأخطاء الملفات ،وواحد لأخطاء الحساب،وهكذا.أو من المحتمل أن يكون لديك معالج خطأ كبير والذي يستخدم عبارات If...Then...Else لتفحص شرط الخطأ المخزن في كائن الخطأ Errالشامل global Err object .

```
ErrorHandler:
If (Err.Number = 5) Then
    ' ----- Handle error-code-5 issues here.
```

تستطيع إيجاد أرقام الأخطاء من أجل الأخطاء المشتركة في مستندات الفيجوال أستوديو،ولكنها معتمدة على أعداد مكتوبة بدقة وغير قابلة للتغير تجعل معالجة الأخطاء الغير تركيبية أقل انتشاراً في هذه الأيام مما كانت عليه قبل الدوت نت. لايهم دائما ما الذي سبب الخطأ.أحياناً،إذا عملت على تمكين معالجة الخطأ حيث كان علي أن لا أمكنه فهذه ليست نهاية العالم،فيما لو وصل الإجراء للنهاية في مسألة تحرير الخطأ،ببساطة أقوم بالتبليغ عن تفاصيل الخطأ للمستخدم،وأتجاوز سطر التوجيه.

```
Public Sub DoSomeWork()
        On Error GoTo ErrorHandler
        ' ----- Logic code goes here.
        Return
ErrorHandler:
        MsgBox("An error occurred in 'DoTheWork':" & Err.Description)
        Resume Next
    End Sub
```

يبلغ مقطع الكود هذا عن الخطأ،ومن ثم يستخدم العبارةResume Next (تنوع للعبارة القياسية Resume)للعودة بسطر الكود مباشرة للسطر الذي يلي السطر الذي سبب الخطأ.يوجد خيار أخر يستخدم (تابع"من عنوان أخر"Resume some_other_label)والتي تعود بالتحكم لمنطقة معينة من الكود من خلال مناطق نوعية مسماة.

تعطيل معالجة الخطأ Disabling Error Handling

استخدام العبارة On Error GoTo يمكّن من معالجة خطأ معين.على الرغم من أنك تستطيع استخدام عبارة On Error GoTo ثانية لإعادة توجيه الأخطاء لمعالج خطأ أخر في إجرائك،الحد الأعلى الذي يبقى خلاله معالج خطأ واحد ساري المفعول هو أي وقت.فحالما تفعل(تمكّن)معالج خطأ ما،فإنه يبقى ساري المفعول حتى نهاية الإجراء،أو تعيد توجيه الأخطاء لمعالج أخر،أو تخصص تبديل إلى غير فعال turn off لمعالجة الخطأ في الروتين(الإجراء).ليأخذ هذا الأخير دوره،أصدر العبارة التالية:

```
On Error GoTo 0
```

تجاهل الأخطاء Ignoring Errors

لايعمل معالج خطأ أي شيء خاص.خذ مقطع معالجة الخطأ التالي:

```
ErrorHandler:
Resume Next
```

عندما يحدث خطأ ما،فهذا المعالج يرجع التحكم مباشرة للسطر الذي يلي السطر الذي سبب الخطأ تماماً.تتضمن الفيجوال بيسك اختصار لهذا الفعل.

```
On Error Resume Next
```

بإصدار العبارة.فإن جميع الأخطاء ستقطن في كائن الخطأ(كما يتم عمله مع جميع الأخطاء،لامشكلة في كيفية معالجتها)ومن ثم يتجاوز السطر الذي سبب الخطأ.ولم يتم إعلام المستخدم عن الخطأ،وسيستمر في استخدام التطبيق مع تجاهل الأخطاء.

معالجة الأخطاء التركيبية Structured Error Handling

كانت معالجة الأخطاء الغير تركيبية مجرد طريقة لمعالجة الأخطاء المتاحة في الفيجوال بيسك قبل الدوت نت.وعلى الرغم من ذلك فقد كانت سهلة الاستخدام،وهي لا تفي بالمتطلبات المفرطة بخصوص إعلان الإصدار2002 للفيجوال بيسك دوت نت أنه نظام برمجة كائنية التوجه.وهكذا،عملت ميكروسوفت على إضافة معالجة الأخطاء التركيبية *structured error handling* للغة،وهي طريقة تستخدم كائنات قياسية للاتصال بالأخطاء،وكود معالجة الخطأ الذي يكون أكثر تكاملاً بإحكام مع الكود الذي يعرضه.
تستخدم هذه الصيغة من معالجة الأخطاء عبارة Try...Catch...Finally المتعددة الأسطر لالتقاط ومعالجة الأخطاء.

```
Try
' ----- Add error-prone code here.
Catch ex As Exception
' ----- Error-handling code here.
Finally
' ----- Cleanup code goes here.
End Try
```

الشرط (التعبير) حاول The Try Clause

تم تصميم عبارة Tryلعرض مقطع أصغر من الكود.وتستطيع أيضاً وضع جميع الكود المصدري لإجراء ما ضمن مقطع Try،وهو أكثر ما يكون مشهور بوضع ضمن ذلك المقطع فقط العبارات التي من المحتمل أن تسبب أخطاء.

```
Try
```

```
    My.Computer.FileSystem.RenameFile(existingFile, newName)
Catch ex As Exception
```

العبارات الآمنة يمكن أن تبقى خارج نطاق الحصة Tryالعبارة Try...End Try. ما يشكل عبارة برمجية آمنة بالضبط هو الموضوع الكثير المناقشة.ولكن نوعين من العبارات هي بشكل عام غير آمنة☹1)تلك العبارات التي تتفاعل مع أنظمة خارجية،مثل ملفات قرص،أو شبكة أو مصادر الأجهزة المركبة على الكمبيوتر،أو حتى المقاطع الكبيرة من الذاكرة،و(2)تلك العبارات التي يمكن أن تؤدي بمتغير ما أو تعبير لأن يتجاوز الحدود المصممة لنوع بيانات ما لذلك المتغير أو التعبير.

التعبير(الشرط) التقط The Catch Clause

يعرف الشرط Catchمعالج خطأ.وكما مع معالجة الخطأ الغير تركيبي،تستطيع تضمين معالج خطأ شامل في عبارة Try، أو تستطيع تضمين معالجات متعددة لأنواع مختلفة من الأخطاء.وكل معالج يتضمن الكلمة المحجوزة Catchالخاصة به.

```
Catch ex As ErrorClass
```

المعرف ex يوفر اسم متغير ما لتفعيل كائن الخطأ الذي تستطيع استخدامه ضمن مقطعCatch. تستطيع منحه أي اسم تريد،وقد يتفاوت من شرط لأخر،ولكن ليس عليه أن يكون كذلك.

تطابق ErrorClassفئة الاستثناءexception: فئة خاصة مصممة بشكل خاص لنقل معلومات الخطأ.فئة الاستثناء الأكثر شمولية هي System.Exception،ويوجد فئات استثناء أخرى أكثر خصوصية تشتق من الفئة System.Exception.بما أن العبارة Try...End Try تنفذ"معالجة موجهة بالكائنات object-oriented error processing" فإن جميع الأخطاء يجب أن يتم تخزينها ككائنات.يتضمن إطار عمل الدوت نت العديد من فئات الاستثناء المسبقة التعريف تم اشتقاقها سابقا من الفئة System.Exception والتي تستطيع استخدامها في تطبيقك.على سبيل المثال، System.DivideByZeroException تلتقط أي خطأ ينشأ (بوضوح) عن تقسيم عدد ما على صفر0.

```
Try
            result = firstNumber / secondNumber
        Catch ex As System.DivideByZeroException
            MsgBox("Divide by zero error.")
        Catch ex As System.OverflowException
            MsgBox("Divide resulting in an overflow.")
        Catch ex As System.Exception
            MsgBox("Some other error occurred.")
        End Try
```

عندما يحدث خطأ ما،يختبر كودك الاستثناء مقابل كل تعبير Catchحتى يجد الفئة المطابقة.يتم تفحص التعابير Catchبالترتيب من الأعلى إلى الأسفل،لذلك تأكد من وضع الأكثر عمومية في الأخير،إذا وضعت في البدايةSystem.Exception، فإن ولا تعبير من تعابير Catchالأخرى في مقطع Tryسيتم إطلاقه أبداً لأن كل استثناء يتطابق معSystem.Exception. إن عدد شروط Catchالتي تضمنها،أو أي استثناءات exceptions تعرضها، يعود إليك.إذا تركت جميع تعابير Catchفارغة،فإنها ستتصرف نوعاً ما مثل العبارةOn Error Resume Next، وإذا ما حدث خطأ ما،فإن جميع العبارات المتبقية في مقطع Tryسيتم تجاوزها.وسيتابع التنفيذ بالمقطع Finally،ومن ثم مع الكود الذي يلي كامل عبارة Try.

الشرط(التعبير) أخيراً The Finally Clause

يمثل الشرط   Finallyجزء من مقطع Tryالذي يقول"اعمل هذا أو مت".إذا ما حدث خطأ ما في عبارةTry، فإن الكود في مقطع Finallyسيتم تنفيذه دائماً بعد أن يتم اكتمال عبارة Catchالمناسبة (ذات الصلة).إذا لم يحدث خطأ،فإن المقطع Finallyسيبقى قيد المعالجة قبل مغادرة العبارة Try. إذا أصدرت العبارة Return في مكان ما في عبارة Try،كذلك سيبقى المقطع Finallyقيد المعالجة قبل مغادرة الروتين(الإجراء).إذا استخدمت العبارة Exit Try للخروج من مقطع Tryمبكراً،فإن المقطع Finallyكذلك سيبقى قيد المعالجة.إذا،بينما يكون مقطع Tryقيد المعالجة،فإن مقطع Finallyسيبقى أيضاً قيد المعالجة.

الشروط Finallyهي اختيارية،لذلك فإنك تضمن فقط شرط Finallyواحد عند الحاجة له.الوقت الوحيد الذي تحتاج فيه لشرط Finallyهو عندما تزيل كل شروط Catch في عبارة Try.

الأخطاء الغير معالجةUnhandled Errors

لقد بينت لك سابقاً في هذا الفصل كيف يمكن للأخطاء الغير معالجة للبيانات الفاسدة أن تقود لانهيار التطبيقات،أو مسارات حلزونية.جميع المبرمجين الجيدين يفهمون أهمية كود معالجة الأخطاء،ويقومون بعمل جهد إضافي لتضمين إما كود معالجة الأخطاء التركيبية أو الغير تركيبية.من المحتمل أن تفكر أن إجراء ما لا يعمل شيء قد يؤدي إلى توليد أخطاء.وسأتركه بدون كود معالجة الخطأ وبالتالي أوفر بعض الوقت في كتابة الكود.هذا ما يخطر ببالك،وبوضوح وبدون أي إنذار:يظهر خطأ ما، وينهار!

عادة،جميع الأخطاء الغير معالجة تظهر تراكم الاستدعاء،تبحث عن الإجراء الذي يتضمن كود معالجة الخطأ.على سبيل المثال،خذ الكود التالي:

```
Private Sub Level1()
        On Error GoTo ErrorHandler
        Level2()
        Return
ErrorHandler:
        MsgBox("Error Handled.")
        Resume Next
    End Sub
    Private Sub Level2()
        Level3()
    End Sub
    Private Sub Level3()
        ' ----- The Err.Raise method forces an
        ' unstructured-style error.
        Err.Raise(1)
```

```
    End Sub
```

عندما يحدث خطأ في المستوى13،يبحث التطبيق عن معالج الخطأ الفعال في الإجراء،ولكن لايجد شيء.لذلك،وبشكل مباشر يخرج من المستوى13 ويعود إلى المستو12،حيث يبحث مرةً أخرى عن معالج الخطأ الفعال.مثل هذا البحث،بكل أسى،سيكون عديم الفائدة.ويدمي القلب،يغادر الكود المستوى12 ونتقل عائداً إلى المستوى11،ويتابع بحثه على معالج حدث معقول.وهذه المرة سيجد واحد.تقفز المعالجة مباشرة إلى مقطع معالج الخطأ ErrorHandler وتنفذ الكود في ذلك المقطع.
إذا كان المستوى11 ليس فيه معالج خطأ،ولايوجد كود يشير إلى التراكم المتضمن على معالج خطأ،فإن المستخدم سيرى نافذة رسالة الخطأ البائسة(راجع الشكل السابق)،متبوعة بموت البرنامج المخيبة Disappointment .
لحسن الحظ،تعمل الفيجوال بيسك على دعم معالج خطأ"التقط الكل catchall" والذي يصطاد مثل هذه الاستثناءات الغير مدارة ويتيح لك فعل شيء ما بخصوصها.وهذه الميزة تعمل فقط إذا كان لديك حقل"تمكين إطار عمل التطبيق Enable application framework" وقد تم اختياره في تبويب التطبيق Application لصفحات خاصيات المشروع project properties.للوصول لقالب كود معالج الأخطاء الشامل،انقر زر"عرض أحداث التطبيق View Application Events" على نفس تبويب خاصيات المشروع.اختر"(أحداث تطبيقي MyApplication Events")من اسم الفئة Class Name للقائمة المنسدلة فوق نافذة الكود المصدري،ومن ثم اختر UnhandledException من قائمة اسم الطريقة Method Name. يظهر الإجراء التالي في نافذة الكود:

```
  Private Sub MyApplication_UnhandledException(ByVal sender As Object, ByVal e As
Microsoft.VisualBasic.ApplicationServices.UnhandledExceptionEventArgs) Handles Me.UnhandledException
  End Sub
```

أضف كود معالجة الخطأ الشامل في هذا الروتين.يتضمن المعامل النسبي e عضو الاستثناء والذي يوفر الوصول لتفاصيل الخطأ بواسطة كائن System.Exception .
إن عضو e.ExitApplication هو خاصية منطقية Boolean والتي تستطيع تعديلها إما للاستمرار أو للخروج من التطبيق.بشكل افتراضي،يتم وضعها إلى صواب،ولذلك عدلها إذا كنت تريد الحفاظ على البرنامج مشتغل.
حتى عندما يبقى البرنامج مشتغل،فإنك ستفقد مسار الحدث النشيط(الفعال)الذي أطلق الخطأ.إذا ظهر الخطأ(أتى) من النقر على زر ما من قبل المستخدم، فإن حدث نقر الزر بالكامل،وكل طرق استدعاءه،سيتم هجرها abandoned مباشرةً وسينتظر البرنامج لإدخال جديد من قبل المستخدم.

إدارة الأخطاء Managing Errors

توجد أشياء أخرى تدور حول الأخطاء يجب أن تتعرف عليها في برامج الفيجوال بيسك.

توليد الأخطاء Generating Errors

صدق أو لاتصدق،توجد حالات من المحتمل أنك تريد فيها توليد أخطاء وقت التنفيذ في كودك.في الحقيقة،العديد من أخطاء وقت التنفيذ التي تصادفها في كودك تحدث بسبب أن ميكروسوفت كتبت كود في مكتبات إطار العمل Framework Class Libraries (FCLs) التي تنتج أخطاء بشكل خاص.وهذا تصميمي.
لنقول أن لديك خاصية لفئة والتي تقبل فقط قيم النسب المئوية من 0 إلى100،ولكن كنوع بيانات عددية صحيحة.

```
Private StoredPercent As Integer
    Public Property InEffectPercent() As Integer
        Get
            Return StoredPercent
        End Get
        Set(ByVal value As Integer)
            StoredPercent = value
        End Set
    End Property
```

قواعدياً لايوجد شيء خطأ في هذا الكود،ولكنه لن يمنع أي شخص من وضع قيمة تخزين مئوية إما 847أو847– وكلاهما خارج نطاق المجال المرغوب.
تستطيع إضافة عبارة الشرط If لمحدد الوصول accessor Set لرفض البيانات المرفوضة،ولكن الخاصيات لا توفر طريقة للعودة بكود الحالات الفاشلة.والطريقة الوحيدة لإعلام الكود المستدعي عن المشكلة هي توليد استثناء.

```
Set(ByVal value As Integer)
             If (value < 0) Or (value > 100) Then
                 Throw New ArgumentOutOfRangeException("value",value, "The allowed range is
from 0 to 100.")
             Else
                 StoredPercent = value
             End If
         End Set
```

والآن،محاولة وضع قيمة للخاصية InEffectPercent خارج المجال 0-to-100 سيولد خطأ،خطأ يمكن أن يتم التقاطه بواسطة معالج الخطأ On Error أو Try...Catch.تقبل العبارة Throw كائن System.Exception أو كائن مشتق منه كمعامل نسبي لها، وترسل كائن الاستثناء ذلك إلى ركام الاستدعاء في البحث عن معالج خطأ ما.
ما يشابه عبارة Throw هي الطريقة Err.Raise.وهي تتيح لك توليد أخطاء باستخدام عدد معتمد على نظام الأخطاء والمعروفة أكثر بالنسبة للفيجوال بيسك6والبيئات الأقدم.إني أوصي باستخدام العبارة Throw،حتى ولو كنت توظف معالجة الأخطاء الغير تركيبية في مكان أخر في كودك.

دمج طرق معالجة الأخطاء Mixing Error-Handling Methods

إنك حر في مزج طرق معالجة الأخطاء التركيبية والغير تركيبية بصورة عامة في تطبيقك،ولكن يمكن أن يستخدم إجراء أو طريقة وحيدة طريقة واحدة فقط. ويمكن أن لاتستخدم كل من On Error و Try...Catch...Finally في نفس الروتين. يمكن لروتين يستخدم On Error أن يستدعي روتين أخر يستخدم Try...Catch...Finally بدون أية مشكلة.
الآن يمكن أنك تفكر بنفسك" أحياناً أستطيع بكل سهولة أن أحدد الوقت الذي أريد استخدام معالجة الأخطاء الغير تركيبية،وأحدد الأوقات الأخرى عندما أرغب باختيار مقاربة تركيبية على الأغلب."وكل هذا يبدو معقولاً جداً،ولكن دعني أحذرك سلفاً أنه توجد معالجة أخطاء متزمتة بحيث تجعلك سخرية لعقود إذا ما استخدمت عبارة

On Error في كودك.لمثل هؤلاء المبرمجين ما هو أساسي عندهم هو التوجه الكائني المجرد،وأي كود يستخدم طرق غير كائنية لتحقيق ما يمكن تحقيقه من خلال محاكاة البرمجة الكائنية التوجه OOPيجب أن يدمر.

رفض عبارة On Error لمثل هذا هو من الغباء بمكان.فكما تذكر من فصول سابقة،أن كل شيء في تطبيق الدوت نت موجه بالكائنات،وبما أن جميع الكود يظهر في سياق كائن ما.فإذا كنت تستخدم معالجة الأخطاء الغير تركيبية،ماتزال تستطيع الحصول على كائن استثناء مناسب من خلال الطريقة ( )Err.GetException،لذلك ليست الكائنات هي القضية الحقيقية.

تقرير متى يتم استخدام معالجة الأخطاء التركيبية أو الغير تركيبية لا تختلف عن ما تم تقريره للاستخدام في السي شارب #Cأو الفيجوال بيسك لكتابة تطبيقاتك.من أجل معظم التطبيقات،المفاضلة في غير محلها.فيمكن أن يكون لدى لغة بعض الميزات الخاصةesoteric بها والتي يمكن أن توجهك في اتجاه ما(مثل المعاملات النسبية الاختيارية للطرق في الفيجوال بيسك)ولكن 99.9 من الميزات المتبقية تكون متشابهة إلى حد ما.

ونفس الشيء صحيح بالنسبة لطرق معالجة الخطأ.وفي بعض الأحيان هناك ميزة سهلة وأفضل من أخرى.على سبيل المثال،خذ الكود التالي والذي يستدعي ثلاث طرق،ولا واحدة منها تتضمن معالج خطأ خاص بها.

```
On Error Resume Next
        RefreshPart1()
        RefreshPart2()
        RefreshPart3()
```

من الواضح،أني لا أبالي فيما إذا كان الخطأ يحدث في واحد من الإجراءات أو لا.إذا ما سبب خطأ ما خروج مبكر من RefreshPart1،الروتين التالي ، RefreshPart2،سيبقى قابل للاستدعاء.وهكذا. غالباً ما احتاج لكود تفحص الخطأ أكثر إتقاناً من هذا.ولكن في كود قليل التأثير،فإن هذا سيكون كافياً.لإتمام نفس الشيء باستخدام معالجة الخطأ التركيبية سيكون هناك تضمين أكثر بقليل.

```
Try
            RefreshPart1()
        Catch
        End Try
        Try
            RefreshPart2()
        Catch
        End Try
        Try
            RefreshPart3()
        Catch
 End Try
```

ذاك الكثير من الكود الزائد بالنسبة لنفس التخصص الوظيفي.إذا كنت من الكارهين لعبارةOn Error .بكل الوسائل استخدم المقطع الثاني من الكود.ولكن إذا كنت مبرمج أكثر عقلانية ،استخدم كل طريقة إذا ما كانت مناسبة في تصميم كوك.

فئة System.Exception نظام.الاستثناء The System.Exception Class

إن الفئة System.Exceptionهي الفئة القاعدية بالنسبة لجميع الاستثناءات التركيبيةstructured exceptions . عندما يحدث خطأ ما.تستطيع تفحص أعضاءها لتحديد طبيعة الخطأ بدقة.وإنك تستخدم أيضاً هذه الفئة (أو واحدة من فئاتها المشتقة)لبناء استثناء خاص بك في انتظار in anticipation استخدام عبارةThrow .

يجدول الجدول التالي أعضاء الكائنSystem.Exception .

عضو الكائنObject member	Description الوصف
Data	خاصية توفر إمكانية الوصول لتجميع أزواج قيم المفاتيح،وكل منها يوفر معلومات إضافية خاصة بالاستثناء.
HelpLink	خاصية تعرف موقع مساعدة على الشبكة يزود بمعلومات متعلقة بهذا الاستثناء.
InnerException	خاصية،إذا كان استثناء ما هو تأثير جانبي عن خطأ أخر،فإن الخطأ الأصلي يظهر هنا.
Message	خاصية ،وصف نصي للخطأ.
Source	خاصية تعرف اسم الاستثناء أو الكائن الذي سبب ذلك الخطأ.
StackTrace	خاصية تعود بنص يوثق بالكامل مسار المقدار الحالي، قائمة بكامل استدعاءات الإجراءات النشيطة التي قادة لعبارات تتسبب بالخطأ.
TargetSite	خاصية تعرف اسم الطريقة التي أطلقت الخطأ.

الفئات المشتقة من System.Exception يمكن أن تتضمن خاصيات إضافية توفر تفاصيل إضافية لنوع خطأ نوعي.

كائن Errالخطأ The Err Object

يوفر الكائن Errإمكانية الوصول لأغلب الأخطاء الحديثة من خلال أعضاءه المتنوعة.أي وقت يحدث خطأ ما،توثق الفيجوال بيسك تفاصيل الخطأ في أعضاء هذا الكائن.على الأغلب يتم الوصول له(لكائن الخطأ) ضمن معالج خطأ غير تركيبي للإشارة أو عرض تفاصيل الخطأ.يبين الجدول التالي أعضاء كائن الخطأ Err.

عضو الكائنObject member	Description الوصف
Clear	طريقة تنظف كل الخاصيات في الكائن،تضع هذه الخاصيات لقيمها الافتراضية.عادة،إنك تستخدم الكائن فقط لتحديد التفاصيل للخطأ الذي تم إطلاقه.ولكن تستطيع أيضاً استخدامه لإدخال خطأ ضمن تفاصيل خطأ خاصة بك.لاحظ شرح الطريقة Raiseفيما بعد هنا.
Erl property	عنوان رقم السطر الأقرب لمكان حدوث الخطأ.في تطبيقات الفيجوال بيسك الحديثة،عناوين الأسطر المرقمة على الأغلب لاتستخدم.لذلك بشكل عام هذا الحقل هو0.
Description property	وصف نصي للخطأ.
HelpContext property	الموضع الذي ضمنه ملف المساعدة على الشبكة فيما يتعلق بالخطأ.إذا كانت هذه الخاصية وخاصية HelpFileتم وضعها،فإن المستخدم يستطيع الوصول لملف معلومات المساعدة ذات الصلة بالموضوع على الشبكة.
LastDLLError	خاصية،القيمة العددية المعادة من معظم الاستدعاء الحديث لـ مكتبة الربط الديناميكي التابعة للدوت نت الأولية،فيما إذا كان خطأ أم لا.
Number property	الكود العددي للخطأ الفعال.

Raise method	استخدم هذه الطريقة لتوليد خطأ وقت التنفيذ.على الرغم من أن هذه الطريقة تعمل على تضمين بعض المعاملات النسبية لوضع خاصيات أخرى في الكائن Err،بإمكانك أيضاً وضع الخاصيات بنفسك قبل استدعاء الطريقةRaise. أي خاصيات تعمل على إعدادها سيتم حفظها في الكائن ليتم اختبارها بواسطة كود معالج الخطأ الذي يستقبل الخطأ.
Source property	اسم التطبيق،أو الفئة،أو الكائن الذي يولد الخطأ الفعال.

الكائن تصحيح The Debug Object

الفيجوال بيسك6(والأقدم)تتضمن أداة مساعدة والتي ستخرج بسرعة معلومات تصحيح من البرنامج،وعرض مثل هذه المخرجات في "النافذة المباشرة Immediate Window " لبيئة تطوير الفيجوال بيسك.

```
Debug.Print("Reached point G in code")
```

إصدار الدوت نت للفيجوال بيسك يحسن كائن التصحيح Debug مع ميزات أكثر،وتغير بسيط في البناء.تم تبديل الطريقة Print بـWriteLine،الطريقة Write تخرج نص بدون لاحقة عودة المشيرة carriage return في النهاية.

```
Debug.WriteLine("Reached point G in code")
```

كل شيء تخرجه باستخدام الطريقة WriteLine (أو ما يشابهها) يذهب إلى سلسلة "المصغيات listeners" "الملحقة (المرفقة) بكائن التصحيحDebug. تستطيع إضافة منصتاتك الخاصة،متضمنة المخرجات لملف العمل.ولكن كائن التصحيح Debugيتم استخدامه فعلياً فقط عند تصحيح برنامجك.حالما تترجم الإصدار الأخير لبرنامجك،فلا تعمل أي من الميزات المتعلقة بالتصحيح بعد ذلك،بشكل تصميمي.

إذا كنت ترغب بتسجيل حالة البيانات من التطبيقات المحررة،خذ استخدام الكائن My.Application.Log (أوMy.Log في برامجASP.NET).فهو مشابه للكائن Debug، يرسل الكائن مخرجاته لأي عدد من المنصتات المسجلةregistered listeners. بشكل افتراضي،جميع المخرجات تذهب لمخرجات تصحيح قياسية(تماماً مثل كائن التصحيحDebug)وإلى ملف التسجيل logfile الذي يتم إنشاءه بشكل خاص لمجمع تطبيقكapplication's assembly .

ميزات أخطاء أخرى للفيجوال بيسكOther Visual Basic Error Features

تتضمن لغة الفيجوال بيسك القليل من العبارات الخاصة بالأخطاء والميزات الأخرى والتي يمكن أن تجد أنها مفيدة:

الدالة (الخطأ إلى نص)ErrorToString(ErrorToString *function*

ترجع هذه الطريقة برسالة الخطأ المرافقة لكود خطأ النظام العدديnumeric system error ،على سبيل المثال،`ErrorToString(10)` تعود بـ"يتم إصلاح هذه المصفوفة أو إغلاقها بشكل مؤقتThis array is fixed or temporarily locked "وهي مفيدة فقط مع أكواد الأخطاء الغير تركيبية الأقدم.

الدالة(هل هو خطأIsError(IsError *function*

عندما تزود معامل نسبي كائني لهذه الخاصية،فإنها تعود بصواب إذا كان الكائن هو كائنSystem.Exception (أو مشتق منه)

مشروع Project

سيكون كود مشروع هذا الفصل مختصر.وسيظهر كود معالجة الخطأ Error-handling على مدى كامل التطبيق،ولكن سنضيفه قليلاً قليلاً كلما تقدمنا في صناعة المشروع.أما الآن،دعنا نركز على إجراءات أو روتينات معالجة الخطأ المركزي والذي سيشغل بعض الفعل الأساسي عندما يحدث خطأ ما في أي مكان في البرنامج. أما بالنسبة لتعابير لامدا، سنؤجل مثل هذا الكود حتى فصل متأخر.

معالج الخطأ العام General Error Handler

بقدر أهمية ودقة الحاجة إلى معالجة الخطأ،فإن تطبيق العمل النموذجي لن يصادف تنوع كبير في أنواع الخطأ.وتطبيقات مثل مشروع المكتبة غير محصنة بشكل رئيسي بالنسبة لثلاثة أنواع من الأخطاء:(1)أخطاء إدخال البيانات،(2)الأخطاء التي تحدث عند قراءة البيانات من أو كتابة البيانات إلى جدول قاعدة بيانات،(3) الأخطاء المرتبطة بالطباعة.من المؤكد أنه توجد أخطاء عددية لتجاوز الحد أو أخطاء أخرى مرتبطة بالبيانات التي قيد الاستخدام،ولكنها على الأغلب تتفاعل مع المصادر الخارجية،مثل قاعدة البيانات والتي هي محور اهتمامنا.

بسبب محدودية أنواع الأخطاء التي تحدث في التطبيق،فمن الممكن كتابة روتين شامل يخبر المستخدم عن الخطأ بأسلوب ثابت(مستقر).كلما حدث خطأ وقت التنفيذ، سنستدعي هذا الروتين المركزي،لنتيح للمستخدم معرفة ما يجري.يمكن لمقطع الكود عندما يحدث الخطأ أن يقرر فيما إذا سيجري أي فعل مكافئ خاص، أو أن يستمر وكأن ليس هناك خطأ قد حدث.

في المشروع،افتح ملف الفئة *General.vb*، وأضف الكود التالي كطريقة جديدة للوحدة البرمجية General.

```
Public Sub GeneralError(ByVal routineName As String, ByVal theError As System.Exception)
        ' ----- Report an error to the user.
        On Error Resume Next
        MsgBox("حدث الخطأ التالي عند الموقع '" & routineName & "':" & vbCrLf & vbCrLf & theError.Message,
MsgBoxStyle.OkOnly Or MsgBoxStyle.Exclamation, ProgramTitle)
        My.Application.Log.WriteException(theError)
    End Sub
```

إليك كيف يعمل.عند تصادف خطأ في روتين ما،يستدعي معالج الخطأ النافذ المفعول الطريقة المركزية GeneralError.

```
Public Sub SomeRoutine()
        On Error GoTo ErrorHandler
        ' ----- Lots of code here.
        Return
ErrorHandler:
        GeneralError("SomeRoutine", Err.GetException())
        Resume Next
    End Sub
```

تستطيع استخدامه مع الأخطاء التركيبية كما يلي:

```
Try
    ' ----- Troubling code here.
Catch ex As System.Exception
```

```
GeneralError("SomeRoutine", ex)
End Try
```

الهدف من الطريقة العامة GeneralError بسيط:ينقل للمستخدم أن هناك خطأ قد حدث،ومن ثم ينتقل للأمام.تستطيع تحسين الروتين ببعض الميزات الإضافية.تسجيل الخطأ إلى ملف خارجي(أو إلى أي منصت مسجل log listener فعال)يمكن أن يساعدك فيما بعد إذا ما احتجت إلى تفحص الأخطاء الناتجة عن التطبيق.كما في السطر التالي المضاف إلى الطريقة GeneralError.

```
        My.Application.Log.WriteException(theError)
```

بالطبع،إذا ما حدث خطأ بينما تتم الكتابة إلى السجل،فذلك سيكون مشكلة كبيرة،لذلك قمنا بإضافة سطر أخر لبداية الطريقة GeneralError.

```
On Error Resume Next
```

التقاط الأخطاء الغير معالجة .Unhandled Error Capture

كما ذكرت سابقاً،إنها فكرة جيدة أن تضمن معالج خطأ شامل في كودك،في حالة إذا ما تغلب خطأ ما على دفاعاتك.لتضمين هذا الكود،أعرض جميع الملفات في مستكشف الحلول باستخدام زر "إظهار جميع الملفات Show All Files"،افتح الملف *ApplicationEvents*.vb، وأضف الكود التالي إلى الفئة MyApplication.

```
Private Sub MyApplication_UnhandledException(ByVal sender As Object, ByVal e As
Microsoft.VisualBasic.ApplicationServices.UnhandledExceptionEventArgs) Handles Me.UnhandledException
            ' ----- Record the error, and keep running.
            e.ExitApplication = False
            GeneralError("Unhandled Exception", e.Exception)
        End Sub
```

بما أن لدينا سابقاً الروتين GeneralError لتسجيل أخطاءنا،من الممكن أن نستفيد منها هنا.

الآدو دوت نت ADO.NET

إذا كنت مطور ويندوز جديد،فإنك لم تتعرف بعد على الاختصارات التالية،بعض الأحيان متعارضة،وأحيان أخرى أدوات تفاعل قاعدة بيانات متممة:

ODBC. فتح قاعدة البيانات في الحالة المتصلة.ODBC—Open Database Connectivity

ISAM. طريقة الوصول التسلسلية المفهرسة.ISAM—Indexed Sequential Access Method

DAO. كائنات الوصول للبيانات.Data Access Objects

RDO.كائنات البيانات البعيدةRemote Data Objects

OLE . ربط وتضمين الكائنات لقاعدة البيانات Object Linking and Embedding for Databases

ADO. تفعيل س لكائنات البيانات. ActiveX Data Objects

عندما تنظر إلى هذه القائمة،من المحتمل أن تفكر(هذا عظيم،يوجد الكثير من الخيارات للاختيار منها.)فستكون أحمق بهذا التفكير.فهذه القائمة ليست عظيمة،إنها مخيفة.تصور،ولدقيقة فقط،فنحن لم نتحدث حول واجهات interfaces قاعدة البيانات،ولكن من ناحية أخرى،هي قضايا أكثر عملية.

عندما قدمت ميكروسوفت تقنية كائن قاعدة البيانات الجديدة ضمن المزيج،عملت على تبعها سريعاً بطفرة إعادة برمجة لجلب "الميراث "الأقدم لتطبيقات للفيجوال بيسك والتي تعود لتقنية قاعدة البيانات الأحدث.وهذا لم يكن ممكناً دائماً،إلى هنا،من الواضح أن آدو دوت نت ،مكتبة قاعدة بيانات ميكروسوفت للدوت نت،مختلف.

إذا كنت على إطلاع على تقنية ADO،تجهز لتنساها.فإن آدو دوت نت ليس بالخليفة الطبيعي لآدوADO .إنه بالكامل تقنية جديدة ليس لها علاقة بآدو،وعلى الرغم من ذلك فهي تتشارك بعض التقنية مع آدو والأدوات الأخرى الأقدم،وتفعل آدو دوت نت هذا لتلعب بعقلك.

ما هو الآدو الدوت نت؟?What Is ADO.NET

إن آدو دوت نت مجموعة من الفئات،المضمنة مع إطار عمل الدوت نت،و تمثل الطريقة الرئيسية التي بواسطتها تتفاعل تطبيقات الدوت نت مع قواعد البيانات العلائقية وأنظمة إدارة وفتح البيانات الخاصة الأخرى ولكنها ليست للتفاعل فقط،في الحقيقة،إن آدو دوت نت قاعدة بيانات علائقية جزئية في الذاكرة قائمة بذاتها.تستطيع إنشاء الجداول والعلاقات(الروابط) من خلال كائنات آدو دوت نت،إضافة ،وإزالة أي سجلات،استعلامات الجداول بالاعتماد على معيار "الاختيار SELECT " وعمل مهمات أخرى بسيطة والتي هي نموذجية بالنسبة لأنظمة قواعد البيانات العلائقية القائمة بذاتها.

جميع الفئات المضمنة مع آدو دوت نت تظهر في فضاء الأسماء System.Data ،بعض فضاءات الأسماء التابعة(الثانوية subordinate)توفر فئات مشتقة مجهزة باتجاه منصات قواعد بيانات خاصة(نوعية)،على سبيل المثال،يستهدف فضاء الأسماء System.Data.SqlClient قواعد بيانات سكولSQL Server databases ، System.Data.OracleClient يركز على أنظمة إدارة قواعد بيانات أوراكلOracle RDBMSsystems ،ويمكن لموفرات قواعد بيانات أخرى تطوير تنفيذات انسيابية لفئات آدو نت المتنوعة للاستخدام مع أنظمتها الخاصة،وتزود بها كفضاءات أسماء منفصلة.

تنفذ ADO.NETتجربة بيانات غير متصلةdata experience *disconnected*. في برمجة قواعد البيانات التقليدية،وخاصة تطبيقات سطح المكتب desktop applications،الاتصال بين تطبيق ما وقاعدة بياناته تم إصلاحه ولأجل طويل.عندما يبدأ التطبيق،يبدأ الاتصال.وعندما يخرج التطبيق بعد عدة ساعات،فإن الاتصال سيتم إنهاءه أخيراً.ولكن في عالم مواقع الانترنت الضخمة والقابلة للتوسع،المحافظة على قاعدة البيانات في وضع الاتصال لساعات طويلة دون انقطاع تكون في بعض الأحيان تبذير(إسراف)وغالباً يكون مستحيل.

تشجعك آدو دوت نت على فتح اتصالات طويلة بشكل كافي للحصول على البيانات التي تفي fulfillsباحتياجاتك المباشرة.حالما تكون البيانات لديك،فإنك تسقط الاتصال حتى تحتاج في الوقت أخر استخراج،أو إدخال،أو تحديث محتوى قاعدة البيانات.إذا أصدرت عبارة سكول التالية:

```
SELECT * FROM Customer WHERE BalanceDue > 0
```

لديك خيار(ا)اختيار جميع السجلات حالاً وبسرعة وبطريقة بسيطة،أو(2)تحميل البيانات إلى جدول في الذاكرة مثل الكائن،وإغلاق الاتصال،والعمل مع السجلات المحملة وكأنها السجلات الأصلية.إذا استخدمت الطريقة الأولى،تستطيع تمضية وقت جيد في التنقل بين السجلات،وتأخذ عدة دقائق في معالجة كل منها.ولكن أدو دوت نت يمتعض (يرفضfrowns upon)هذا النوع من التصرف الأناني.الهدف هو الدخول وأخذ البيانات بالسرعة الممكنة.بسبب الطبيعة الغير متصلة disconnected nature للدوت نت،فبعض التقنيات الشائعة في تطبيقات قواعد البيانات تحتاج التغير.على سبيل المثال،البحث طويل الأمد في سجلات قاعدة البيانات خلال تعديل المستخدم ("التزامن السيئpessimistic concurrency ")صعب التحقيق في البيئة الغير متصلة للدوت نت.سيكون عليك استخدام طرق أخرى،مثل التعاملات(المداولات) أو ميزات الإجراء المخزن الصغيرة،لإنجاز نفس الهدف.

نظرة شاملة عن الدوت نت Overview of ADO.NET

يقسم الدوت نت عالمه إلى نصفي كرة:الموفرات *providers*ومجموعة البيانات*data set* .توفر الموفرات إمكانية الوصول إلى محتوى ما،مثل قاعدة بيانات أوراكل . مجموعة البيانات تعمل على تحضير وجلب المحتوى الأصلي الذي تم الحصول عليه من المخزن الطويل الأمد(قاعدة البيانات).حالما يتم الإحضار،فإما سيتم استهلاكه أو سيتم إعادته إلى قاعدة البيانات للتخزين الطويل الأمد.إذاً تمنحك الموفرات إمكانية الوصول إلى البيانات المخزنة،بعض البيانات يمكن أن يتم نقلها ومعالجتها من خلال تطبيق ما ومجموعة بياناته على قاعدة تخزين قصيرة الأجل.

الموفرات Providers

أنظمة قواعد البيانات الضخمة database systems،مثل مخدم سكول SQL Server وأوراكلOracle ، هي مخدمات قائمة بذاتها تتفاعل مع أدوات وتطبيقات العميل بشكل غير مباشر فقط.هذه الأنظمة تقبل بشكل عام اتصال شبكة من العملاء من خلال منفذ بروتوكول التحكم بالإرسال/بروتوكول الانترنت TCP/IP port أو اتصال مشابه.حالما يتم التصديقauthenticated، يستطيع الزبون عمل جميع متطلباته من خلال هذا الاتصال قبل قطع الاتصال عن النظام.في عام 1990نفذت ميكروسوفت (فتح قاعدة البيانات في الحالة المتصلة*ODBC*)(بالاعتماد على معايير موجودة)كنظام عام من خلاله يمكن للعميل أن يتصل بمخدمات قاعدة البيانات،بالإضافة لمصادر البيانات الأخرى الأبسط.لن يقلق العملاء بعد هذا حول جميع بروتوكولات شبكة العمل الضرورية لمحاكاة قاعدة البيانات،كل ذلك الكود تم تضمينه في مشغل ODBC.

أطلقت ميكروسوفت فيما بعد نظام اتصال بيانات مشابه يدعى*OLE DB* (ربط وتضمين الكائنات لقاعدة البيانات)،بالاعتماد على تقنية ActiveX.حالما ظهرت مشغلات OLE DB للأنظمة المشتركة،وعلى الرغم من أنك ماتزال تستطيع الدخول إلى مصادر ODBCمن خلال مشغل ODBCالشامل المبني ضمن OLE DB.

في الدوت نت،كل من ODBCو OLE DB تم استبدالهما بكود المكتبات libraries، والموفرات *providers* التي توفر جميع الاتصالات بين قاعدة البيانات وتطبيقك.إن الموفرات هي الجزء المكمل لآدو دوت نتADO.NET، وعليك استخدامها للحصول على قواعد بياناتك.لحسن الحظ،تتواجد الموفرات من أجل أنظمة قواعد البيانات الرئيسية،ويتواجد موفر OLE DB من أجل الأنظمة التي هي بدون موفر بنفسها.

من أجل الكائنات الرئيسية التي تكّون وجهة نظر المبرمج عن الموفر:

كائن الاتصال ***The* Connection *object***

هذه الكائن يوجه الاتصال بين برنامجك ومصدر البيانات.إنه يتضمن خاصيات وطرق تتيح لك الإشارة إلى موقع أو وسيطات الاتصال لمصدر البيانات.العديد من أوامر العمليات(الإجراءات) يتم إدارتها عند مستوى هذه الكائن.

كائن الأمر ***The* Command *object***

يأخذ هذا الكائن عبارة سكول SQLالتي توفرها،ويحضّرها ليتم نقلها من خلال كائن الاتصال.تستطيع تضمين وسيطات في أمرك لتخزين إجراء ودعم عبارة معقدة.

كائن قارئ البيانات ***The* DataReader *object***

يوفر كائن قارئ البيانات طريقة بسيطة وفعالة لاستخراج نتائج من استعلام بيانات .تستخدمه كائنات أخرى في الدوت نت لاستقبال وإعادة توجيه البيانات للاستخدام ضمن برنامجك،ولكن يستطيع كودك استخدامه مباشرةً لمعالجة نتائج عبارة "اختر SELECT "أو إجراء استخراجي لبيانات أخرى.

كائن محول البيانات ***The* DataAdapter *object***

هذا الكائن هو مايجعل الاتصال بين مجموعة البيانات وباقي الموفرات ممكنة.واحد من أعماله الرئيسية هو تعديل عبارات معالجة البيانات(عبارات SELECT،و INSERT،و UPDATE،و DELETE)المولدة بواسطة مجموعة البيانات في تنسيق يمكن أن يتم استخدامه بواسطة مصدر البيانات المناسب(ذو الصلة).

استخدام هذا الكائنات معقد قليلاً،ولكنها ليست صعبة الفهم. للاتصال بقاعدة بيانات علاقية نموذجية،مثل مخدم سكول،ومعالجة البيانات،اتبع الخطوات التالية:

1.أسس اتصال لمصدر بياناتك باستخدام كائن الاتصال.

2.ضمن عبارة سكول في كائن الأمر.

3.نفذ كائن الأمر في سياق الاتصال المؤسس له.

4.إذا كان يجب استعادة أي بيانات،استخدم إما قارئ البيانات DataReaderلعمل مسح خلال السجلات،أو دمج(ضم)لكل من "محول بيانات DataAdapter " و"مجموعة بيانات DataSet (أو جدول بياناتDataTable)" لاستخراج أو تخزين النتائج.

5.أغلق كل الكائنات التي قمت بفتحها لمعالجة البيانات.

على الرغم من أن إطار عمل الدوت نت يتضمن موفرات بيانات للعديد من أنظمة البيانات المختلفة،فإن باقي المناقشة في هذا الفصل ستركز على موفر مخدم سكول فقط، والذي يتم عرضه من خلال فضاء الأسماء System.Data.SqlClient.

ملاحظة :تتضمن سكول سرفر2005 دعم لميزة تدعى"حالات المستخدمUser Instances "للاستخدام مع قواعد بيانات سكول سرفر2005 بطبعته السريعة.تتيح هذه الميزة امتياز أدنى low-privilege للمستخدم للوصول لملف قاعدة بيانات سكول سرفر السريع دون الحاجة لمدير administrator من أجل تأسيس إعدادات أمان سكول سرفر لذلك المستخدم.وهذه الميزة مفيدة في البيئات حيث البرمجيات المرتبطة تم تنصيبها من خلال طريقة التوزيع "بنقرة واحدة *ClickOnce* "(سيتم مناقشتها في الفصل 25 إن شاء الله)دون تضمين للمدير administrator .ويتطلب أيضاً تركيب خاص لتنصيب لسكول سرفر السريع قبل الاستخدام.ولمزيد من المعلومات حول هذه الميزة راجع موضوع"العمل مع حالات المستخدم"في قسم "الدوت نتWorking with User Instances " في مستندات شبكة مطوري ميكروسوفتMSDN .والموفرة مع تنصيب الفيجوال أستوديو إذا كنت قد نصبت هذه المستنداتMSDN .

مجموعات البيانات Data Sets

إذا كنت في طريقك لعمل أكثر من مسح scanللبيانات الراجعة من استعلام قارئ البياناتDataReader، من المحتمل أنك سوف تستخدم مجموعة البيانات data set لتخزين أو إدارة أو بشكل اختياري تحديث بياناتك.توفر كل مجموعة بيانات عرض غير متصل(بدون اتصال)شامل generic disconnected view للبيانات،سواء كانت بيانات من موفر أو بيانات عملت على بناءها من خلال الكود .على الرغم من أن كل موفر يتم ربطه بمنصة قاعدة بيانات معينة(مثل أوراكل)أو اتصال قياسي(مثل OLE DB)،الكائنات في مملكة مجموعة البيانات هي شاملة،ويمكن التفاعل معها بأي موفرات خاصة بمنصة.

ثلاث كائنات تركب عالم مجموعات البياناتdata sets :

كائن مجموع البيانات *The* DataSet *object*

يتصرف كل كائن مجموعة بيانات كقاعدة بيانات صغيرة.تستطيع إضافة العديد من الجداول لمجموعة البيانات كما ترغب،وتؤسس علاقات مفاتيح ثانوية foreign-key relationships بين حقول fieldsهذه الجداولtables. ذاتيات internalsكل مجموعة بيانات هي غامضة لايمكن سبرها unfathomable mystery، ولكن تستطيع تصدير كامل مجموعة بيانات DataSetلـ XML. وتعيد تحميله مرةً أخرى فيما بعد لما كان عليه إذا كان يجب عليك ذلك.

كائن جدول البيانات ***The* DataTable *object***

كل جدول tableفي مجموعة بياناتك DataSetيستخدم كائن جدول بيانات Data Table منفصل،وقابلية الوصول له تتم من خلال تجميع جداول Tables collection مجموعة البيانات DataSet .إن جداول البيانات مفيدة أيضاً ككائنات قائمة بذاتها.إذا خططت لإضافة جدول مفرد لمجموعة بياناتكDataSet، من المحتمل أنك تفضل استخدام فقط كائن جدول بيانات DataTableلوحده بدون مجموعة بياناتDataSet .ضمن كل كائن جدول بياناتDataTable، كائنات عمود بيانات DataColumn وصف بيانات DataRowمنفصلة تؤسس تعريفات حقل fieldوقيم values بيانات حقيقية،على الترتيب.

كائن علاقة البيانات ***The* DataRelation *object***

استخدم كائنات علاقة البياناتDataRelation، المخزنة ضمن تجميع علاقات مجموعة البياناتDataSet's Relations collection، لتأسيس علاقات على مستوى الحقل field-level relationships وقيود constraintsبين الأعمدة في كائنات جدول البياناتDataTable .

على الرغم من أن مجموعات البيانات data sets على الأغلب يتم استخدامها مع الموفراتproviders، تستطيع استخدامها بشكل منفصل لبناء تجميع خاص بك في الذاكرة للجداول والعلاقات. وهذا مشابه لـ"مجموعات سجلات العميلclient-side record sets " والتي تستطيع بناءها بكائنات آدو السابقة لآدو دوت نت

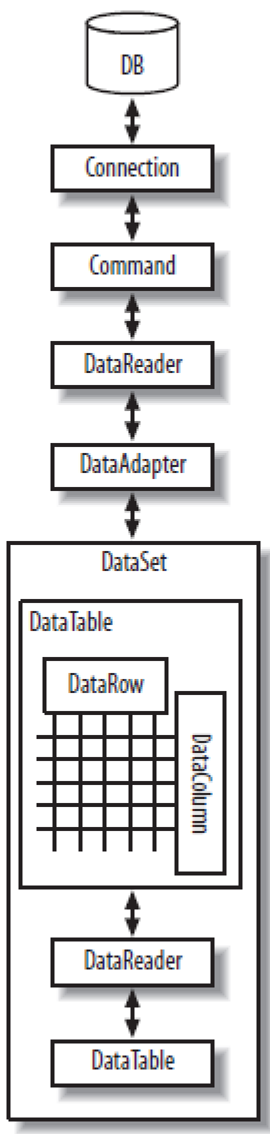
على الرغم من الميزات المضمنة في ADO.NET التي تجعل مجموعات البيانات data sets أكثر قوة من مجموعات السجلات الأقدم pre-.NET ADO، older record sets.

ملاحظة:تتضمن الفيجوال بيسك2008 ميزة"مجموعات بيانات محددة النوعTyped DataSets" تستخدم لتكامل مجموعة بيانات DataSetمع بيانات معينة أو تنسيق سجلrecord format. تستخدم تقنية لينكو LINQالجديدة ميزة مشابهة للمساعدة في تأسيس علاقات بين لينكو وجداول قاعدة البياناتdatabase tables .

مجموعات البيانات مقابل بدون مجموعات بيانات Data Sets Versus No Data Sets

عندما يتم استخدام كل من الموفرات ومجموعات البيانات مع بعضها فإنها تمنح واجهة نهاية(طرف) – إلى- نهاية(طرف) لقيم البيانات المستقلة من حقول في جداول قاعدة البيانات إلى بنود في الذاكرة لسجل صف بيانات DataRow. يبين الشكل التالي تفاعل هذه الكائنات.عندما تتفاعل مع بيانات من قاعدة بيانات خارجية،إنك دائماً تستخدم فئات الموفرprovider ،ولكن يعود لك فيما إذا كنت تريد استخدام مجموعات البياناتdata sets. يوجد محاسن وسيئات لكلا الطريقتين،بعض منها يظهر في الجدول التالي:

بدون مجموعات البيانات Without data sets	مع مجموعات البياناتWith data sets
عليك توفير جميع عبارات سكول،في تنسيق يتوقعه الموفر.وهذا صحيح لجميع متطلبات SELECT و INSERT و UPDATE و DELETE.	يعمل كل من مجموعة البيانات DataSetومحول البيانات DataAdapterمع بعضهما لصنع العديد من عبارات سكول عوضاً عنك.
البيانات المستخرجة من خلال DataReaderهي للقراءة فقط.عليك أن تصدر أوامر منفصلة لتحديث البيانات.	قراءة البيانات من قاعدة البيانات يمكن أن يتم تعديلها في الذاكرة ،ويتم تحديثها كمجموعة مع استدعاء طريقة مفردة.
تنقلات البيانات فعالة جداً،بما أنه ليس هناك حاجة لتكاليف نقل البيانات ضمن تركيب مجموعة بيانات معقد.	يمكن أن يكون هناك أداء ناجح بما أن مجموعة البيانات تبني الكائنات الضرورية المطلوبة لكل سجل منقول.
تجميع الذاكرة يكون محدود لسجل مفرد مساوي لحقول البيانات،زائد بعض التكاليف الصغيرة.	تجميع الذاكرة مطلوب لمجموعة ناتجة بالكامل،زائد التكاليف بالنسبة لكل جدول،حقل، عمود،والصفوف في المجموعة الناتجة.
يمكن أن يتم فتح قارئ بيانات DataReaderمفرد فقط في نفس الوقت(مالم يدعم الموفر MARS والذي سأناقشه فيما بعد)	أي عدد من مجموعات البيانات يمكن أن تكون قيد الاستخدام في نفس الوقت.
عمر الاتصال بقاعدة البيانات يتواجد طالما أن قارئ البيانات DataReaderقيد الاستخدام. فإذا استغرقت خمس دقائق لمسح مجموعة بيانات ناتجة لأنك تعمل الكثير من تحليل السجلات المسبقة،فإن الاتصال سيبقى فعال كامل الخمس دقائق.	اتصالات البيانات يتم حفظها فقط بطول كافي لنقل البيانات من أو إلى قاعدة البيانات فقط .
تحضر قارئات البيانات DataReadersسجل كل مرة.ويجب أن يتم معالجة السجلات في ترتيب الاستخراج.	تستطيع القفز فوق السجلات في مجموعة البيانات،وتمايز بينهم لتفي بمتطلباتك.
تمضي الكثير من الوقت في العمل مع النصوص(من أجل عبارات سكول وحقول بيانات الصف.	جميع حقول البيانات يتم تنظيمها بشكل منطقي،تماماً مثل ما كانت في قاعدة البيانات الفعلية. تستطيع التفاعل معها بشكل مباشر.
كل أمر Commandواتصال Connectionيعمل مع مصدر بيانات يدعم موفر مفرد.	يمكن أن تتصل جداول البيانات DataTablesالمختلفة ضمن مجموعة بيانات DataSetإلى مصادر بيانات متمايزة (مختلفةdistinct data sources).وأيضاً،تستطيع صنع البيانات يدوياً لذلك كل صف بيانات DataRowيحتوي بيانات من مصادر مختلفة.
بسبب أنك أنت الذي تدير جميع عبارات سكولSQL، فإن لديك (نسبياً relatively) مستوى عالي من السيطرة على عمليات تفاعل البيانات بالكامل.	بسبب أن عرض البيانات هو مجرد abstracted، فإن لديك (نسبياً)مستوى محدود للسيطرة على عملية تفاعل البيانات بالكامل(على الرغم من أن الاستخدام المتقدم لمجموعات البيانات يعمل على منحك بعض التحكم الإضافي).



الموفرات ومجموعات البيانات أثناء العمل Providers and data sets in action

بالنسبة لي فإن المدخلة الأخيرة في الجدول السابق هي الحاسمة.إن عمل المبرمج هو السيطرة على تجربة برمجيات المستخدم،والأكثر تحكماً هو الأفضل.ولهذا السبب فإني أكره المعالجات السحرية wizards و"مولدات الكود code generators " والتي تأخذ التحكم عوضاً عني،أو عوضاً عن المطور.على الرغم من أنني أصبر على القالب المزود بواسطة الفيجوال أستوديو عند إنشاء مشروع جديد.مايزال وكما سترى ،لمحات التحكم الشخصية على كودي في مشروع المكتبة،مع اعتمادي الكبير على قارئ البيانات DataReadersأكثر من مجموعات البيانات DataSets. عندما أعمل على تخزين البيانات على المدى الطويل،فإني عادة ألصق البيانات فقط في جدول بيانات DataTableدون تضمين مجموعة بيانات DataSet .

دعم مجموعات ناتجة عن نشاط متعدد. MARS Support

لقد ذكرت فيما سبق شيء ما يدعى MARSو MARSاختصار"مجموعات ناتجة عن نشاط متعدد Multiple Active Result Sets" عادة،يسمح كائن اتصال Connectionمفرد لقارئ بيانات DataReaderمفرد لأن يكون قيد الاستخدام في وقت معين.وهذا التحديد هو ثنائي الاتجاه bidirectional. إذا كنت تعمل مسح قارئ بيانات باستخدام عبارة "أختر SELECT " فلا تستطيع إصدار عبارات إدراج INSERT،تحديث UPDATE، أو حذف DELETEعلى نفس الاتصال حتى تغلق قارئ البيانات DataReader. مع تقديم MARS، يمكن لاتصال وحيد الآن معاملة نشاطات انتقال بيانات في أي اتجاه.أضافت سكول سرفر دعم MARSمع الإصدر2005. وكذلك دعمت أراكل MARS كالميزات منذ الإصدار الأول للدوت نت.تبدو اتصالات MARS كميزة تحتاج لتمكينها باستمرار.ولكنها تعمل على إضافة تكاليف إضافية لتطبيق تبطئ تطبيقك.وأيضاً لا تمتزج دائماً مع التطبيقات ذات مسارات التنفيذ المتعددة multithreaded applications .

الاتصال إلى سكول سرفر باستخدام فيجوال أستوديو. Connecting to SQL Server with Visual Studio

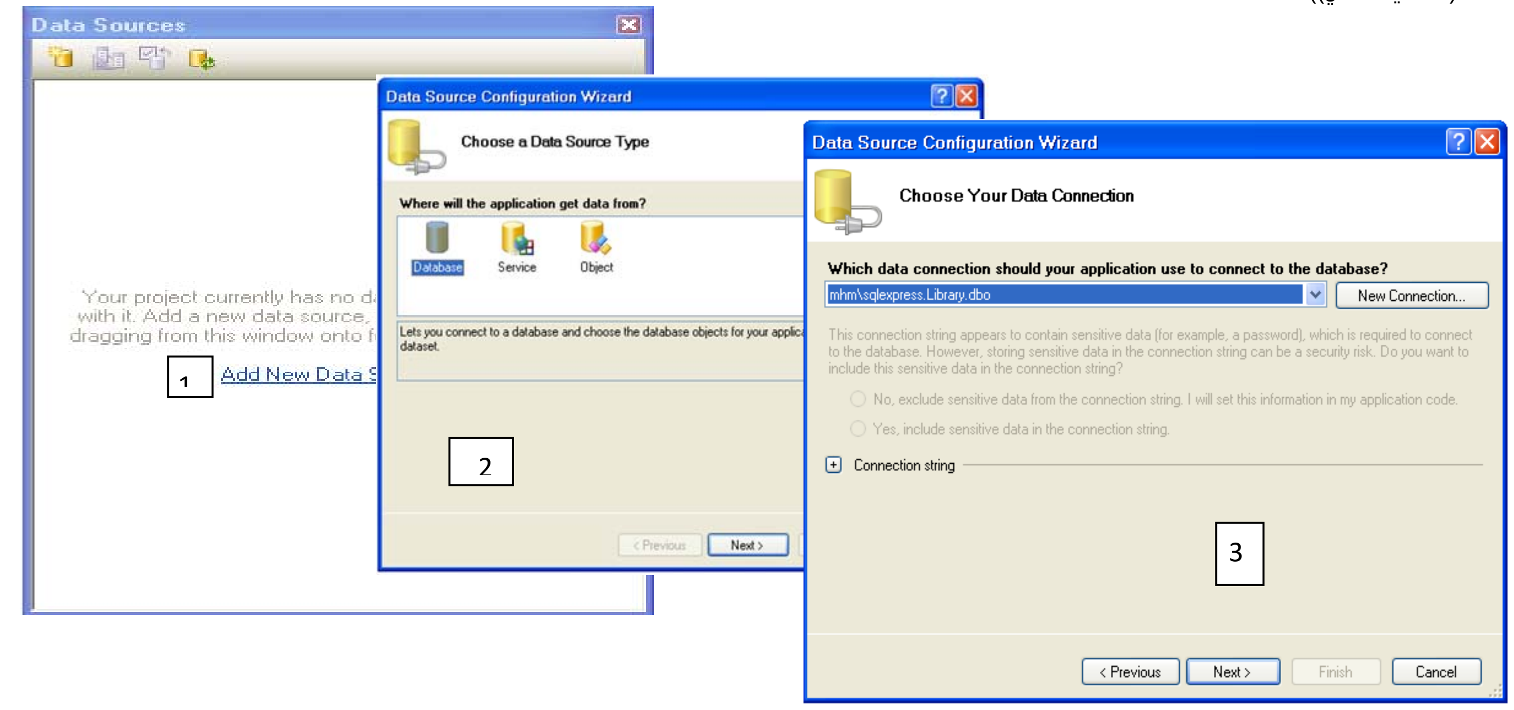
إن للفيجوال أستوديو العديد من الأدوات الجاهزة والتي تجعل العمل مع البيانات بسيط كبساطة السحب والإسقاط.حسناً،ليس بهذه السرعة.ولكن بالإجابة على القليل من الأسئلة وسحب وإسقاط بند واحد،تستطيع بناء تطبيق كامل يتيح لك تحديث البيانات في قاعدة بياناتك.لنجرب ذلك معاً.

إنشاء مصدر بيانات Creating a Data Source

ابدأ مشروع نماذج ويندوز جديد في الفيجوال أستوديو،أختر بيانات Data >> أمر قائمة "أظهر مصادر البيانات Show Data Sources " مما يحضر لوحة مصادر البيانات،كما هو مبين في الشكل التالي.

إن المشروع الجديد لا يتضمن أي مصدر بيانات بشكل افتراضي.لذلك تحتاج لإضافة مصدر بيانات.أنقر على الوصلة"أضف مصدر بيانات جديد Add New Data Source " في لوحة " مصادر البياناتData Sources ".يقودك"المعالج السحري لتركيب مصدر بيانات" خلال عمليات إنشاء مصدر بيانات:

1.تسألك الخطوة الأولى"من أين سيحصل التطبيق على البيانات؟اختر "قاعدة البياناتDatabase " وانقر الزر "التالي Next "(كما هو مبين في الشكل التالي (التخطيط الثاني)).



2.تسأل الخطوة التالية،"أي اتصال بيانات يجب أن يستخدمه تطبيقك للاتصال بقاعدة البيانات؟"سيكون علينا إنشاء اتصال جديد لقاعدة بيانات المكتبة والتي عملنا على تصميمها في الفصل الرابع.انقر زر "اتصال جديدNew Connection "(المبين في الشكل السابق (التخطيط الثالث))

3.يظهر حوار اختيار مصدر البيانات.اختر ميكروسوفت سكول سرفر Microsoft SQL Server من قائمة"مصدر البياناتData source "،ومن ثم انقر زر "استمر Continue ".إذا كنت قد وصلت لهذا الحوار من قبل، واخترت حقل الاختيار"دائماً استخدم هذا الاختيارAlways use this selection "من المحتمل أن لايظهر هذا الحوار على الإطلاق.

4.يظهر حوار إضافة اتصال Add Connection لتجميع تفاصيل الاتصال الجديد.. إذا كان حقل"مصدر البياناتData source "يحتوي على شيء ما غير "ميكروسوفت سكول سرفرMicrosoft SQL Server "انقر على زر"تغيير Change " لتبديل نوع الاتصال باستخدام الحوار المذكور في الخطوة الثالثة.

5.العودة إلى نموذج"إضافة اتصال" وقد تم ملئ حقل"اسم المخدمServer name "باسم نسخة سكول سرفر SQL Server instance خاصتك.هذه القائمة المنسدلة لديها نسخ مجدولة لأسماء المخدمات،ولكن إذا لم تكن كذلك،عليك إدخالها بنفسك.الاسم الافتراضي بالنسبة لسكول سرفر السريع هو اسم كمبيوترك،مع اللاحقة"SQLEXPRESS\ ".فإذا كان اسم كمبيوترك هو " MYSYSTEM "فاسم النسخة سيكون " MYSYSTEM\SQLEXPRESS."(كما مبين في الشكل السابق (التخطيط 3 فاسم نسختي mhm\ SQLEXPRESS)).

6.ركب إعدادات التصديق(التوثيق)في مقطع"الدخول إلى المخدمLog on to the server ".بالنسبة لي فإني استخدم توثيق ويندوز القياسي،ولكنه يعتمد على كيفية تثبيت قاعدة البيانات في الفصل الرابع.

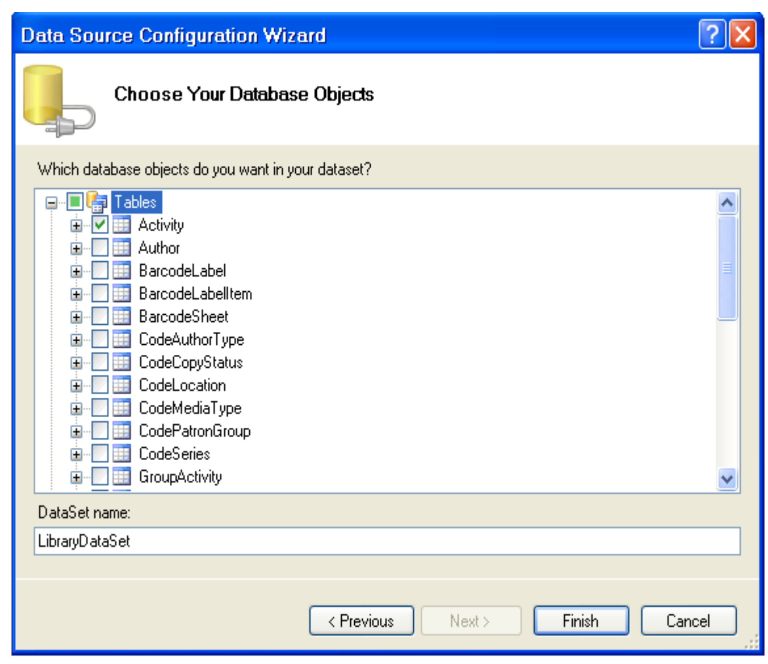
7.في مقطع "اتصال إلى قاعدة البياناتConnect to a database "،إما اختر أو اكتب " **Library** "من أجل اسم قاعدة البيانات.

8.انقر على زر "اختبر الاتصالTest Connection " للتأكد من أن العمل كله صحيح.عندما تنتهي انقر على زر "موافق OK " لإنشاء الاتصال الجديد..

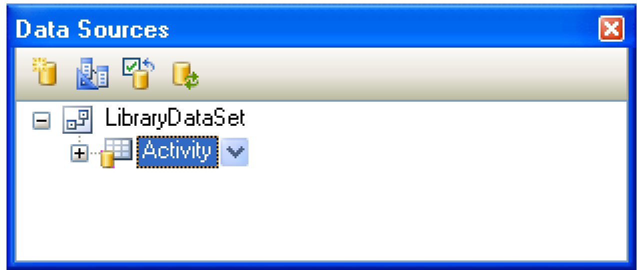
9.بعد النقر على زر "موافق OK " سيتم إعادتنا إلى نموذج المعالج السحري لتركيب مصدر البيانات.والاتصال الذي عملنا على إنشاءه يجب أن يظهر الآن في القائمة كما يظهر في الشكل السابق(التخطيط الثالث).

10.اللوحة التالية تسأل فيما إذا سيصبح مصدر البيانات هذا جزء من الإعدادات القابلة للتركيب من أجل هذا المشروع this data source should become part of the configurable settings for this project.سندخل في ميزات الإعدادات الخاصة بالفيجوال بيسك في الفصل 14.ولكن الآن ،عليك فقط قبول الافتراضي والنقر على "التاليNext ".

11.هذه هي الخطوة الأخيرة في إنشاء مصدر البيانات.ترى اللوحة النهائية قائمة بميزات البيانات الناتجة في قاعدة بيانات المكتبة.افتح تفرع الجداول Tablesواختر " Activity "كما هو مبين في الشكل التالي.ومن ثم انقر "إنهاءFinish ".

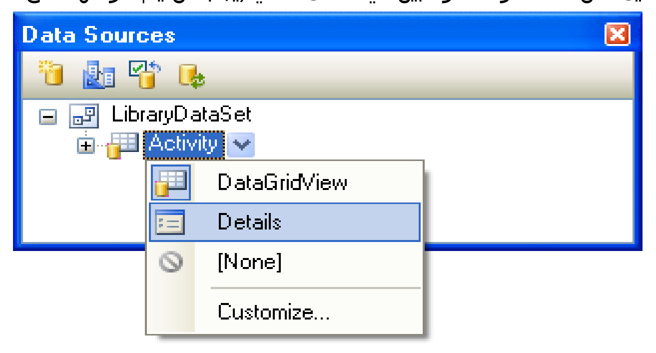


راجع لوحة مصادر البيانات المبينة في الشكل التالي.فهي تتضمن على مصدر بيانات "LibraryDataSet"مع وصلته إلى جدول " Activity".

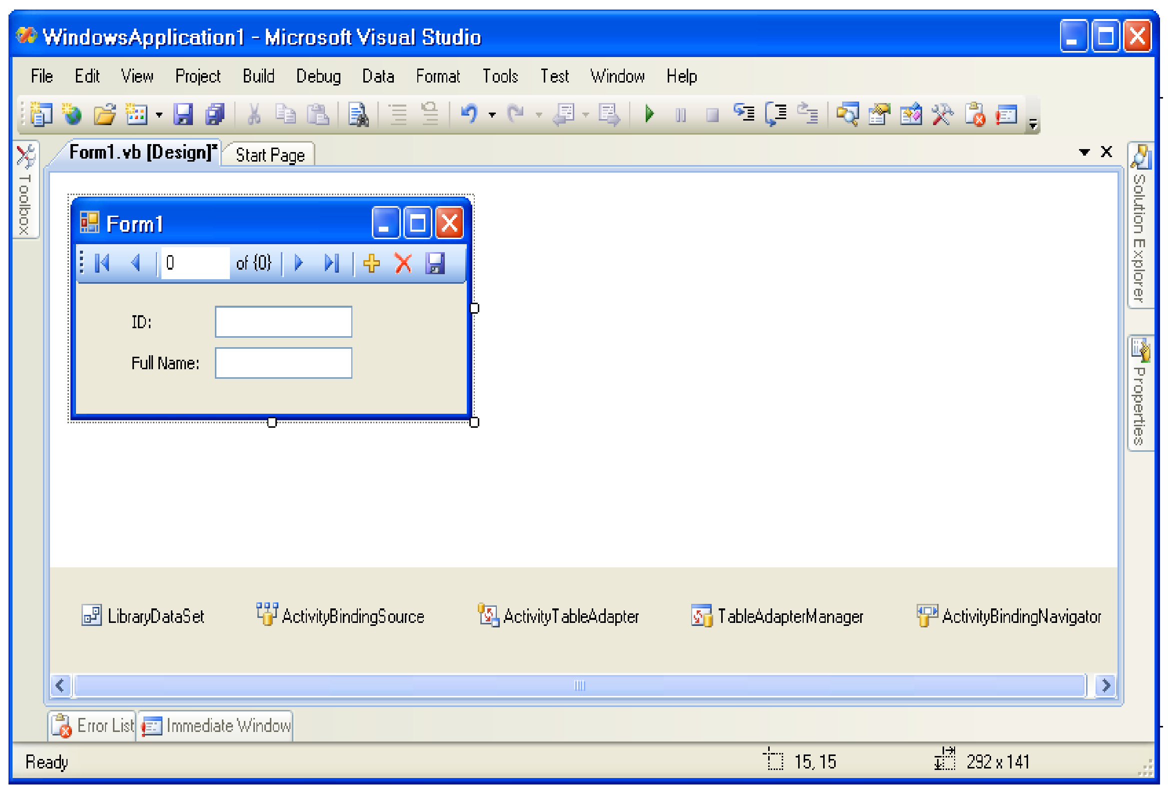


استخدام مصدر بيانات Using a Data Source

على أية حال ما هو مصدر البيانات؟إنه ببساطة وصلة إلى قسم في قاعدة بياناتك،تم إنهاءه في كائن دوت نت نموذجي.والآن هو جزء من مشروعك وبإمكانك استخدامه للوصول للبيانات في جدول " Activity" من خلال كود مشروعك،أو بواسطة "السحب والإسقاطdrag-and-drop " .في لوحة مصدر البيانات،ستجد المدخلة" Activity" ولها قائمة منسدلة فعلية.اختر "تفاصيل" من القائمة،وكما هو مبين في الشكل التالي.(يجب أن يتم عرض سطح الفورم1 Form1من أجل هذا العمل).



أخيراً،اسحب واسقط مدخلة Activity على سطح Form1.وعندما الاستمرار،ستضيف الفيجوال أستوديو مجموعة من الأدوات للفورم،زائد العديد من الأدوات التي ليس لديها واجهة مستخدم non-userinterface تحت الفورم تماماً.(شاهد الشكل التالي).



بمجرد السحب والإسقاط،تضيف الفيجوال أستوديو جميع الأدوات الضرورية والوصلات ليدور نموذجك ضمن التوربين المشحون لمحرر جدول " Activity".جربه الآن بالضغط على المفتاح F5.في البرنامج المشتغل،تستطيع استخدام تحكم الوصول للسجلات كما في تخطيط برنامج ميكروسوفت أكسس وذلك للتنقل بين السجلات لجدول " Activity"(الأسهم في أعلى الفورم).تستطيع أيضاً تعديل القيم لكل سجل،وإضافة سجلات جديدة،أو حذف سجلات موجودة(ولكن أعد تخزين جميع الأشياء لما كانت عليه،حيث أننا سنستخدمها فيما بعد).تكلم عن القوة ، البساطة،الأسطر غير الموظفة !فمن يريد أن يدفع لمبرمجين مثلنا ،عندما تستطيع الفيجوال أستوديو فعل كل هذا لك؟

تحزيم البيانات Data Binding

في الحقيقة،إن كل ما تعمله الفيجوال أستوديو ليس بالكثير.حيث أنها تستخدم ميزة تدعى"تحزيم البياناتData binding " لربط حقول الفورم مع مصدر البيانات(جدول قاعدة بيانات المكتبة).إن تحزيم البيانات هو ميزة جاهزة ضمن أدوات نماذج ويندوز تسمح لهذه الأدوات وبشكل آلي عرض وتحرير القيم من مصدر البيانات المصاحب، مثل قاعدة البيانات.ويتم تصنيف(فرز)الجميع من خلال خاصيات الأداة.اختر أداة FullNameTextBox المضافة لنموذج هذا المشروع،ومن ثم تفحص خاصياتها.في الأعلى تماماً توجد خاصية مسماة"حزم البياناتDataBindings " وهي خاصية نصية جزئية تحتوي " ActivityBindingSource – FullName"مشيرةً إلى الأداة ActivityBindingSource التي ليس لها واجهة مستخدم والتي تم إضافتها بواسطة الفيجوال أستوديو وبشكل آلي. والأداةActivityBindingSource، في دورها،تحتوي مرجع إلى الكائن LibraryDataSet (مصدر البيانات data source الذي عملنا على إنشاءه سابقاً).يتصل مصدر البيانات إلى مخدم سكول سرفر SQL Server، ومنه إلى قاعدة بيانات المكتبة Library database وأخيراً إلى الجدول Activityوحقله FullName.

إذا عملت على إحصاء جميع الكائنات المضمنة في علاقة تحزيم البيانات هذه ،تصل إلى ما يقارب 5،238كائن متمايز.لاتتعجب من كون الفيجوال أستوديو تقوم بعمل الكثير لأجلك.يوفر تحزيم البيانات الكثير من الراحة،ولكنه يأخذ منك الكثير من التحكم والتطوير.على الرغم من أن هناك خاصيات وأحداث تتيح لك إدارة سمات تحزيم البيانات وعمليات التحديث الخاصة بها،فإن معظم الكود الأساسي يكون مخفي بعيداً داخل أقسام تحزيم البيانات للدوت نت.يمكن أن لا تمسه،أو تشوهه،ومجرد كونه كذلك فإنه سيء.فمعتقدي:البرمجيات الجيدة تتضمن تحكم(سيطرة)أعظمية للمطور،وتحكم أقل بالنسبة للمستخدم.

كجزء من عملك كمطور توفير بيئة مصنوعة بشكل متقن للمستخدم من أجل الوصول للبيانات الهامة.وهذه المتطلبات التي لديك تتحكم بتجربة المستخدم من خلال كودك المصدري.على وجه التخصيص،ستؤجل الكثير من ذلك التحكم إلى تحكم آخر عندما تستخدم أدوات أخرى خارجية.طالما أن هذه الأدوات تسمح لك بالتحكم بمعرفة المستخدم لمستوى تكون راض عنه.هذا عظيم.ولكنني دائماً أشعر بالإحباط من تحزيم البيانات،ما عدا وقت تنفيذ عرض بيانات للقراءة فقط من قاعدة البيانات.ميزات مشابهة كانت في الفيجوال بيسك لوقت طويل وقبل أن تصل الدوت نت،وهذه الميزات قد تم جعلها صعبة بالنسبة للمطور من أجل التحكم بتفاعلات البيانات المتنوعة على الفورم.

لحسن الحظ،إذا كنت تتجنب ميزات تحزيم البيانات،فإن الفيجوال بيسك ستمرر لك مسؤولية إدارة جميع التفاعلات بين قاعدة البيانات والمستخدم.

التفاعل مع سكول سرفر في الكود.Interacting with SQL Server in Code

الاتصال مع قاعدة البيانات بنفسك هو بالتأكيد أكثر من عمل سحب وإسقاط مصادر البيانات.

بناء نص الاتصال Building the Connection String

الخطوة الأولى على الطريق إلى نمط حياة التحكم بالبيانات هو الاتصال إلى قاعدة البيانات باستخدام نص الاتصال.إذا كنت قد استخدمتADO، فإنك على معرفة مسبقة بنصوص الاتصال المستخدمة في ADO.NET،بشكل عام نصوص الاتصال connection strings متشابهة.من المحتمل أيضاً أنك تعرف أنه من خلال نصوص الاتصال تحفظ ميكروسوفت لجام محكم tight rein على مطوري ويندوز.وهي ليست بهذا التعقيد،وهي لاشيء أكثر من نصوص وسيطات مفصولة بفاصلة منقوطة.ولكن الوسيطات parametersالتي يجب أن تضمن،وتنسيقها الدقيق هو قوام عنوان تفسيري stuff of legend . مستندات ميكروسوفت المضمنة MSDNمع الفيجوال

أستوديو توفر بعض الأمثلة لنصوص الاتصال،ولكن ليس مع الكثير من التفاصيل.يوفر مصدر ثالث *ttp://www.connectionstrings.com* أيضا عدد ضخم من الأمثلة عن تنسيقات نصوص اتصال صحيحة.

نص الاتصال الذي سنستخدمه للاتصال إلى قاعدة بيانات المكتبة،ولحسن الحظ ليس معقد بإفراط.إذا استخدمت تسجيل دخول ميكروسوفت ويندوز الخاص بك للاتصال إلى قاعدة البيانات،النص التالي سيوفر احتياجاتك(كسطر واحد غير مفصول)

Data Source=*instance_name*;Initial Catalog=Library;Integrated Security=true

بالنسبة لنص الاتصال الذي عملت على بناءه للاتصال مع قاعدة بيانات المكتبة هو التالي:

Data Source=MHM\SQLEXPRESS;Initial Catalog=Library;Integrated Security=True

حيث يتم استبدال *instance_name* باسم بنسخة مخدم سكول الخاص بك أو مصدر البيانات،نص الاتصال الذي عملنا على بناءه سابقاً يستخدم "MYSYSTEM\SQLEXPRESS" كاسم مصدر بياناته.

لاستخدام معرف مستخدم سكول سرفر SQL Server user IDs وكلمة المرور passwords،جرب التنسيق التالي:

Data Source=*instance_name*;Initial Catalog=Library;User ID=*sa*;Password=*xyz*

بالطبع،استبدل معرف المستخدمuser ID (*sa*)وكلمة المرور password (*xyz*) بإعداداتك الخاصة.إذا كنت تريد تضمين دعم MARSفي اتصالك،أضف مكونات فاصلة منقوطة أخرى:

MultipleActiveResultSets=true

ملاحظة:خيارات أخرى لنص الاتصال Other connection string options تتيح لك الاتصال إلى ملف قاعدة بيانات مخدم سكول سرفر السريع SQL Server Express (SSE)بشكل مباشر،غير طريقة"نسخة المستخدم user instancing"(التي تستخدم غالباً مع قواعد بيانات التوزيع بنقرة واحدة ClickOnce-deployed databases)،وأعمل التعديلات الأخرى،على الرغم من أنها مبعثرة نوعاً ما،تستطيع إيجاد هذه الخيارات موثقة في مستندات MSDNالتي تأتي مع الفيجوال أستوديو.

تأسيس الاتصال.Establishing the Connection

استخدم نص الاتصال لإنشاء كائنSqlConnection، ومن ثم افتح الاتصال.أنشئ تطبيق نماذج ويندوز جديد في الفيجوال أستوديو وأضف أداة زر لسطح الفورم. انقر مزدوج على هذا الزر للوصول إلى معالج حدث النقر.ومن ثم أضف الكود التالي لمعالج الحدث هذا.

```
' ----- Assumes:
        ' Imports System.Data
        Dim libraryDB As New SqlClient.SqlConnection("Data Source=mhm\SQLEXPRESS;" & "Initial Catalog=Library;Integrated Security=true")
        libraryDB.Open()
```

تأكد من تبديل " mhm"باسم النظام الخاص بك، فلاسم mhm المبين هو اسم النظام الخاص بي.إن مقطع الكود الكامل هذا يبدو سهل جداً وأسهل بالنسبة لي من الخطوات العشر أو الخمسة عشرة والتي عليك إتباعها سابقاً عند إعداد وتركيب الاتصال خلال الفيجوال أستوديو.

استخدام عبارات سكول Using SQL Statements

حالما يتم فتح الاتصال،تستطيع إصدار "اخترSELECT"،" تحديثUPDATE"،" حذفDELETE"،"إدراج INSERT "أو أية عبارة من عبارات لغة معالجة البيانات data manipulation language (DML)أو أية عبارة من عبارات لغة تعريف البيانات data definition language (DDL)المقبولة بواسطة قاعدة البيانات.

يحضر الكائن SqlCommandعبارة سكول SQLالخاصة بك ليتم استخدامها بواسطة الاتصال المفتوح.إليك عبارة تعود بوصف المدخلة رقم 1 في جدول Activity.

SELECT FullName FROM Activity WHERE ID = 1

إنشاء كائن SqlCommandوالذي يدور حول هذه العبارة سهل.إن المشيد لكائن SqlCommandيأخذ عبارة سكول SQL،زائد كائن SqlConnection.

أضف الكود التالي لنهاية معالج حدث نقر الزر السابق:

```
Dim sqlStatement As New SqlClient.SqlCommand("SELECT FullName FROM Activity WHERE ID = 1", libraryDB)
```

معالجة النتائج Processing the Results

الشيء الوحيد الباقي والذي عليك عمله هو تمرير عبارة سكول SQLلقاعدة البيانات،بواسطة الاتصال،واستخراج النتائج ككائن SqlDataReader.حالما تحصل على البيانات،عالج كل سجل باستخدام طريقة الكائن Read. إنك تصل إلى الحقول المستقلة بواسطة الاسم من خلال تجميع البنود Item الافتراضي.أضف هذا الكود الإضافي لنهاية معالج حدث نقر الزر السابق:

```
Dim sqlResults As SqlClient.SqlDataReader = sqlStatement.ExecuteReader()
        sqlResults.Read()
        MsgBox(CStr(sqlResults.Item("FullName")))
        ' أن بما Item أيضاً يعمل التالي فإن الافتراضية الخاصية هو
        ' MsgBox(CStr(sqlResults("FullName")))
        ' والاختصار التالي يعمل أيضاً
        ' MsgBox(CStr(sqlResults!FullName))
```

مع أخذ كل مقاطع الكود السابقة مع بعضها يعرض الرسالة المبينة في الشكل التالي.



عندما تنتهي،تأكد من أنك أغلقت جميع الاتصالات التي قمت بفتحها.لذلك أضف المقطع الصغير التالي من الكود إلى معالج حدث نقر الزر السابق:

```
sqlResults.Close()
```

```
libraryDB.Close()
```

تعديل البيانات Modifying Data

إن إجراء تغيرات إلى جداول قاعدة البيانات يتم بكتابة الكود تماماً مثل استخراج البيانات،ولكن ليس هناك حاجة "لقارئ بيانات سكول".وبدلاً من استخدام الطريقة ExecuteReader،استخدم الطريقة ExecuteNonQuery والتي لا تعود بنتائج.

```
Dim sqlStatement As New SqlClient.SqlCommand("UPDATE Activity SET FullName = 'Sleeps all day'" & "
WHERE ID = 1", libraryDB)
sqlStatement.ExecuteNonQuery()
```

إن للسكول سرفر2005 ميزة مريحة بحيث تعود بحقل مفرد من سجل جديد منشئ بواسطة العبارة "إدراج INSERT ".إذا راجعت تصميم قاعدة بيانات مشروع المكتبة،سترى أن الحقول IDفي عدة جداول يتم توليدها بشكل آلي.
تقليدياً،إذا أردت استخراج حقل IDبشكل مباشر إلى سجل جديد،عليك أولاً إدراج INSERTالسجل ،ومن ثم إجراء عبارة "اختر SELECT"منفصلة، العودة بحقل ID لسجل جديد:.

```
INSERT INTO CodeSeries (FullName)
VALUES ('Children''s Books')
SELECT ID FROM CodeSeries
WHERE FullName = 'Children''s Books'
```

مخرجات سكول سرفر المدرجة تضم كل من هاتين العبارتين في إجراء وحيد :

```
INSERT INTO CodeSeries (FullName)
```

المخرجات المدرجة لحقل **ID**.

```
VALUES ('Children''s Books')
```

عندما يكتمل الإدراجINSERT، يعمل سكول سرفر على إرجاع حقل IDكمجموعة نتيجةresult set، تماماً كأنك أصدرت عبارة "اختر SELECT"منفصلة.إن طريقة ExecuteScalar لكائن أمر سكولSqlCommand هي طريقة بسيطة لاستخراج قيمة مفردة من استعلام سكول.

```
sqlStatement = New SqlClient.SqlCommand("INSERT INTO CodeSeries (FullName) " & "OUTPUT INSERTED.ID VALUES
('Children''s Books')",libraryDB)
        Dim newID As Integer = CInt(sqlStatement.ExecuteScalar())
```

مداولات قاعدة البيانات Database Transactions

يمكّن إجراء المداولات *Transactions* "الجميع أو لاشيء"عبر عبارات سكول متعددة.حال البدء،إما يتم إصدار جميع عبارات سكول ضمن سياق المداولة المكتملة،أو لايكتمل أي منها.إذا كان لديك 10 تحديثات updatesلبيانات يجب أن يتم عملها،ولكن قاعدة البيانات فشلت بخمسة منهم،تستطيع التراجع roll back عن المداولة بالكامل.تعكس reversesقاعدة البيانات العبارات السابقة،وتعيد تخزين البيانات لما كانت عليه قبل أن تبدأ المداولة.(ولكن تحديثات من مستخدم أخر لا تتأثر بالتراجع).إذا نجحت جميع العبارات،تستطيع تنفيذ *commit*المداولة transaction بالكامل،وجعل كل تغيرات المداولة ثابتةpermanent. بالنسبة لقواعد بيانات سكول سرفر SQL Server databases،يتم إدارة المداولات من خلال موفر الكائنSqlTransaction. مثل ميزات ADO.NET الأخرى فهو سهل الاستخدام.تبدأ المداولة باستدعاء الطريقة BeginTransactionعلى الاتصالconnection .

```
Public atomicSet As SqlClient.SqlTransaction = libraryDB.BeginTransaction()
```

لتضمين عبارة سكول في المداولة،أسند كائن SqlTransactionإلى خاصية "المداولة Transaction " للكائن SqlCommand.

```
sqlStatement.Transaction = atomicSet
```

ومن ثم استدعي الطريقة المناسبة على الأمر.عندما تكتمل جميع الأوامر،استخدم الطريقة Commitللمداولةtransaction لجعل التغيرات ثابت permanent .

```
atomicSet.Commit()
```

إذا،بدلاً من الإدراج،احتجت إلى إجهاض المداولةtransaction، استخدم طريقة "التراجع Rollback "

```
atomicSet.Rollback( )
```

كينونة آدو دوت نت لإطار العمل.ADO.NET Entity Framework

الإصدار ADO.NET المضمن مع النسخة 3.5من إطار عمل الدوت نت(وهي النسخة التي تأتي مع النسخة الحالية أي فيجوال أستوديو2008)تتضمن مكونات جديدة:كينونة إطار عمل أدو دو نتADO.NET Entity Framework . جزء أداة نمذجة علاقة الكينونةPart entity-relationship modeling tool،جزء مولد الكودpart code generator،وهذه التقنية الجديدة تساعدك على صنع عروض بيانات منطقة logical data views للبيانات المخزنة في قاعدة البيانات العلائقية الخاصة بك أو مصدر البيانات.
تتيح لك كينونة إطار العمل Entity Framework تصميم ثلاث أنواع من الكائنات بالاستناد على بياناتك المخزنة:الكينونات *entities* (مشابهة للجداول)،العلاقات *relationships*(روابط قاعدة البيانات)،و مجموعات الكينونة *entity sets* (الكينونات المتعلقة(ذات الصلة)).كل نوع يتم تمثيله بواسطة كائنات تعرض أعضاء قاعدة بياناتك الجوهرية(القاعدية) في صيغة أكثر برمجيةin a more programmable fashion. على سبيل المثال، تستطيع تصميم كينونة كائن تمثل جدول في قاعدة بياناتك،وأعضاء كائن يمكن أن تمثل الحقول fieldsفي سجل وحيد single record.إنها تبدو مثل بعض الميزات المبنية ضمن كائن جدول البيانات DataTableلآدو دو نتADO.NET، إن ما يجعل كينونة إطار العمل مفيدة جداً هو التالي☹1)فهو يشبك البيانات الفعلية بالعروضviews، (2)إنه دعم لوراثة الكينونة entity inheritance،(3)ولها المقدرة على التصرف كـموفر ADO.NET.

تشبيك البيانات *Data mapping*

تشبيك البيانات مشابه لإنشاء"عرض view " في قاعدة بيانات علائقية تقليدية،تستطيع إنشاء كينونات والتي يتم تركيبها ببعضها من امتداد سجلات مصادر متعددة عبر جداول قاعدة بيانات متعددةmultiple database tables. وهذا يتضمن عروض أباء-أبناء للبياناتparent-child views،تستطيع إنشاء كينونة "طلبية Order " والتي تشير إلى كل من سجل الطلبية orderالرئيسي وبنود سطر "الطلبية Order" المتعددة المضمنة في الطلبية.منطقياً،ويتم معاملة هذه الكينونة الجديدة كعنصر استعلامي مفرد. عندما تطلب بيانات من خلال الكينونة الجديدة،ليس عليك تعليمها كيفية ربط وإقامة علاقة سببية ومنطقية بين القطع المصدرية المتنوعة للبيانات كل مرة.

وراثة الكينونة *Entity inheritance*

حالما يكون لديك كينونة معرفة،تستطيع تمديد تركيب الكينونة من خلال الوراثة.مثلاً،من المحتمل أنك تريد إنشاء كينونة تدعى InternalOrder معتمدة على كينونة"الطلبية Order"الأصلية، وإضافة أعضاء بحيث تقتفي أثر البيانات المخصصة لتلبيات داخلية.يمكن لهذه الحقول الجديدة أن تكون جدول مخصص،يتم إطلاقه بواسطة إشعار منطقي Boolean على الجدول الرئيسي للطلبيات.ولكن هذا ليس بالقضية المهمة:من المستطاع إخفاء الجميع بواسطة منطق الكينونة نفسها.عندما تطلب حالة(نسخة)من InternalOrder،فإنها ستعرف تماماً أنك تعني فقط النوع المشار إليه والمخصص داخلياً للطلبية،وليس الطلبيات القياسية.

دعم مزود آدو دوت نت *ADO.NET provider support*

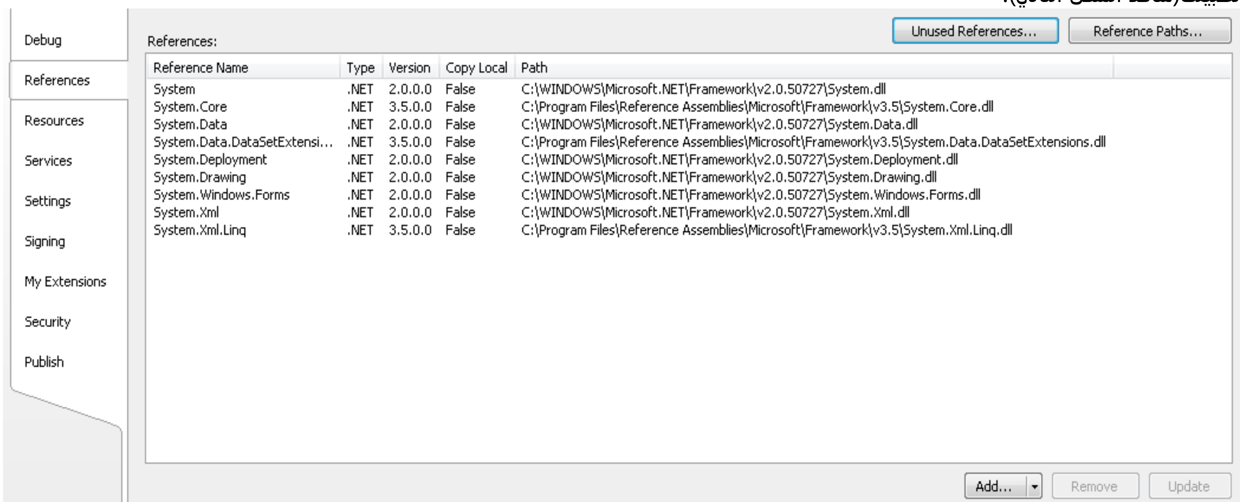
حالما تكون قد أنشأت كينوناتك ومنطق التشبيك المتعلق بها،تستطيع "الاتصال"إلى الكينونات وكأنه تم تخزينها في قاعدة بياناتك الخاصة.بدلاً من الاتصال إلى سكول سرفر والاستعلام عن الجداول مباشرةً،فإنك تتصل إلى السياق المشبك والاستعلام على عرض المنطق الجديد لبياناتك.
بعض هذه التقنية والتي سأناقشها في الفصل 17لديها صيغة قريبة للكود المركز على كينونة إطار العمل للدوت نت ،ولكن وبشكل خاص ستستهدف بشكل خاص سكول سرفر.إن كينونة إطار العمل لآدو دوت نت هي الطريقة الرئيسية لاتصال تقنية لينكو الجديدة للدوت نت إلى قواعد بيانات غير التابعة لميكروسوفت مثل أوراكل و DB2.
لسوء الحظ،قبل عدة أشهر من إطلاق فيجوال أستوديو2008،أعلنت ميكروسوفت أن كينونة إطار العمل Entity Framework لن تكون جاهزة في الوقت الحالي من أجل المنتجات الرئيسية(مثل أوراكل).فقد تم نقدها لبزوغها كإصدار منفصل بعد عدة أشهر من إصدار الفيجوال أستوديو.ويمكن أن يكون أو لايكون المنتج النهائي متاح عند قراءتك لهذه الأسطر.

مشروع Project

من المحتمل أن أكثر من 50% من كود مشروع المكتبة سيتضمن الوصول المباشر لقاعدة البيانات،أو معالجة البيانات المستخرجة من خلال ADO.NET.باستمرار إنشاء كائنات أمر وقارئ بيانات جديدة،على الرغم من البساطة،ولكن صعبة نوعاً ما على الأصابع.بما أن الكثير من الكود يتم تكراره،فإن الكود في مشروع هذا الفصل سيحاول التركيز على بعض الأساسيات،والكود المعياريboilerplate .

مرجع فضاءات أسماء البيانات Reference the Data Namespaces

لكود مشروع المكتبة مراجع لعدة فضاءات أسماء هامة في الدوت نت مثل System،وSystem.Windows.Forms و Microsoft.VisualBasic.مهما يكن،ليس له حتى الآن مرجع لأي من فضاءات أسماء آدو دوت نتADO.NET . (تذكر أن "التزويد بمرجعreferencing " يعني إمكانية الوصول إلى مكتبة الربط الديناميكي للدوت نتNET DLL. في المشروع واستخدام أنواعها في كودك) قبل استخدام فضاءات الأسماء هذه في الكود،نحتاج إلى إنشاء مراجع(أو مؤشرات)لها.وهذا يتم عمله من خلال نافذة خاصيات المشروعproject properties، في تبويب"المراجع References ".سترى قائمة بالمجمعات التي تم الإشارة إليها كمراجع سابقاً في تطبيقك(شاهد الشكل التالي).



لإضافة مرجع جديد ،انقر الزر "إضافة Add "الموجود تحت القائمة مباشرةً،واختر مرجع،فإذا طلب منك نوع المرجع الذي سيضاف.على نموذج إضافة مرجع،سيكون تبويب دوت نت فعال سابقاً.ومن المدهش رؤية كيف تم تثبيت العديد من مجمعات الدوت نت على نظامك سابقاً.ولكن لا تجلس وتحدق كالأبله،اختر كل من System.Data و System.Xml من قائمة المكونات،ومن ثم انقر الزر موافقOK. وبالتالي ستتضمن قائمة المراجع الآن في خاصيات المشروع كل مكتبات فضاء الأسماء المختارة.
نستطيع الآن الإشارة إلى فئات في فضاء الأسماء System.Data مباشرةً.ولكن كتابة System.Data قبل كل استخدام لفئة ذات صلة بالبيانات data-related class سيسبب الملل.نستطيع نشر العبارات "Imports System.Data" لجميع الملفات في المشروع،ولكن توفر الفيجوال أستوديو حل أكثر مركزية.بما أنه ما يزال لديك تبويب المراجع Referencesمفتوح،انظر للأسفل إلى مقطع "فضاءات الأسماء المجلوبةImported namespaces ". تشير قائمة الاختيار الضخمة إلى أي من فضاءات الأسماء سيتم إحضارها بشكل آلي على كامل تطبيقك.وفضاءات الأسماء هذه لا تحتاج إلى عبارات Imports مفصولة في كودك،ولكن يتصرف كودك وكأنك قد عملت على إضافتهم على أية حال.والآن ارجع إلى الأعلى واختر صندوق الاختبار بجانب مدخلة System.Data في هذه القائمة.ومن ثم أغلق نافذة خاصيات المشروع project properties.
معظم الكود الجديد لهذا الفصل سيظهر في الملف*General.vb*، لذلك افتحه الآن.سنستخدم متغيران على مستوى المشروع(عام) لإدارة اتصال قاعدة بيانات رئيسية إلى قاعدة بيانات المكتبة.المتغير الأولLibraryDB، وهو كائن SqlConnection يستخدم نص اتصالنا لقاعدة connection string بيانات المكتبة.الكائن

HoldTransaction ذو الصلة بالمتغير السابق سيحفظ SqlTransaction عندما تكون المعاملة قيد التنفيذ.أضف هذين السطرين إلى الوحدة البرمجية .لقد عملت على وضعهما قبل الطريقة الموجودة CenterText تماماً.

```
Public LibraryDB As System.Data.SqlClient.SqlConnection
Public HoldTransaction As System.Data.SqlClient.SqlTransaction
```

الاتصال بقاعدة البيانات Connecting to the Database

بما أن مشروع المكتبة سيعتمد كثيراً على قاعدة البيانات،سنبني كائن SqlConnection عندما يبدأ التطبيق بالتشغيل .

ملاحظة:

الاحتفاظ بالاتصال على طول التطبيق يأتي ضد النصيحة التي قدمتها سابقاً أن اتصال قاعدة البيانات يجب أن يكون قصير الأمد.مهما يكن ،من أجل الحفاظ على الكود بسيط قدر الإمكان من أجل أهداف التوضيح التدريبية،اخترت هذه المقاربة.وأيضاً لأن مشروع المكتبة تم تصميمه من أجل قاعدة صغيرة،فليس هناك حاجة لأن يكون قابل للتوسيع إلى حدٍ كبير.

يحتوي الإجراء ConnectDatabase جميع الكود الضروري لإنشاء هذا الكائن.أما الآن،فلقد عملت على كتابة فقط نص الاتصال connection string في هذا الروتين.في الفصول اللاحقة سنعمل على تضمين معلومات الاتصال كجزء من نظام الإعداد.أضف الروتين التالي إلى الوحدة البرمجية General .وتأكد من تغير المرجع(المؤشر MYSYSTEM) لما هو ضروري على نظامك الخاص كما ذكرنا سابقاً.

```
Public Function ConnectDatabase() As Boolean
        ' ----- Connect to the database. Return True on success.
        Dim connectionString As String
        ' ----- Initialize.
        HoldTransaction = Nothing
        ' ----- Build the connection string.
        ' !!! WARNING: Hardcoded for now.
        connectionString = "Data Source=MYSYSTEM\SQLEXPRESS;" & "Initial Catalog=Library;Integrated 
Security=true"
        ' ----- Attempt to open the database.
        Try
            LibraryDB = New SqlClient.SqlConnection(connectionString)
            LibraryDB.Open()
        Catch ex As Exception
            GeneralError("ConnectDatabase", ex)
            Return False
        End Try
        ' ----- Success.
        Return True
    End Function
```

هذا الروتين هو الروتين الرئيسي للمشروع وهو حدث التطبيق العملي MyApplication_Startup، من ملف الكود المصدري لـ *ApplicationEvents.vb*،من أجل بناء كائن الاتصال لدى بداية تشغيل التطبيق،أضف الكود التالي لنهاية معالج الحدث هذا.

```
Private Sub MyApplication_Startup(ByVal sender As Object, ByVal e As 
Microsoft.VisualBasic.ApplicationServices.StartupEventArgs) Handles Me.Startup
            ' ----- Connect to the database.
            If (ConnectDatabase() = False) Then
                Me.HideSplashScreen()
                e.Cancel = True
                Return
            End If
        End Sub
```

عندما يخرج المستخدم من تطبيق المكتبة،فإن الكود سيستدعي الطريقة CleanUpProgram للتخلص بشكل مناسب من كائن الاتصال.ارجع إلى الوحدة البرمجية *General.vb* وأضف تلك الطريقة.

```
Public Sub CleanUpProgram()
         ' ------ استعد للخروج من البرنامج.
         On Error Resume Next
         LibraryDB.Close()
    End Sub
```

لتبسيط الأشياء،نستدعي هذا الروتين من معالج حدث التطبيق MyApplication_Shutdown، ارجع إلى ملف *ApplicationEvents.vb* وأضف التالي:

```
   Private Sub MyApplication_Shutdown(ByVal sender As Object, ByVal e As System.EventArgs) Handles 
Me.Shutdown
            ' ----- أغلق قاعدة البيانات.
            CleanUpProgram()
        End Sub
```

التفاعل مع قاعدة البيانات.Interacting with the Database

بعد أن تم تأسيس اتصال قاعدة البيانات،حان الوقت لعمل شيء ما به.تنفذ الروتينات الأربع المتمركزة في البداية الكثير من الكود الذي تم مناقشته سابقاً:إنشاء قارئات بيانات data readers وجداول tables، ومعالجة كود سكول SQL العام.أضف هذه الروتينات إلى الوحدة البرمجية كما يلي:

```
    Public Function CreateDataTable(ByVal sqlText As String) As Data.DataTable
        ' ------ تقديم عبارة سكول للرجوع بجدول بيانات.
        Dim dbCommand As SqlClient.SqlCommand
        Dim dbAdapter As SqlClient.SqlDataAdapter
        Dim dbTable As Data.DataTable

        ' ----- محاولة تشغيل العبارة
 'لاحظ أنه لايوجد عملية اصطياد للأخطاء هنا،ويرجع للروتين المستدعي تثبيت اختبار الأخطاء.'
        dbCommand = New SqlClient.SqlCommand(sqlText, LibraryDB)
        If Not (HoldTransaction Is Nothing) Then dbCommand.Transaction = HoldTransaction
        dbAdapter = New SqlClient.SqlDataAdapter(dbCommand)
        dbTable = New Data.DataTable
        dbAdapter.Fill(dbTable)
        dbAdapter = Nothing
        dbCommand = Nothing
        Return dbTable
    End Function

    Public Function CreateReader(ByVal sqlText As String) As SqlClient.SqlDataReader
        ' ------ تقديم عبارة سكول للرجوع بقارئ بيانات
        Dim dbCommand As SqlClient.SqlCommand
        Dim dbScan As SqlClient.SqlDataReader

'محاولة تشغيل العبارة،لاحظ أنه يتم إصطياد الأخطاء هنا.
'وتثبيت إختبار الخطأ راجع إلى الروتين المستدعي.
        dbCommand = New SqlClient.SqlCommand(sqlText, LibraryDB)
        If Not (HoldTransaction Is Nothing) Then dbCommand.Transaction = HoldTransaction
        dbScan = dbCommand.ExecuteReader()
        dbCommand = Nothing
        Return dbScan
    End Function

    Public Sub ExecuteSQL(ByVal sqlText As String)
'تقديم عبارة سكول لتشغيلها.
        Dim DBCommand As SqlClient.SqlCommand

'محاولة تشغيل العبارة.لاحظ أنه لايوجد إصطياد أخطاء هنا.
'وتثبيت اختبار الخطأ راجع إلى الروتين المستدعي.
        DBCommand = New SqlClient.SqlCommand(sqlText, LibraryDB)
        If Not (HoldTransaction Is Nothing) Then DBCommand.Transaction = HoldTransaction
        DBCommand.ExecuteNonQuery()
    End Sub

    Public Function ExecuteSQLReturn(ByVal sqlText As String) As Object
'تقديم عبارة سكول،تشغيلها،والعودة بالنتيجة المناسبة
        Dim DBCommand As SqlClient.SqlCommand

'محاولة تشغيل العبارة.لاحظ أنه لايوجد إصطياد للأخطاء هنا
'وتثبيت إختبار الخطأ راجع إلى الروتين المستدعي.
        DBCommand = New SqlClient.SqlCommand(sqlText, LibraryDB)
        If Not (HoldTransaction Is Nothing) Then DBCommand.Transaction = HoldTransaction
        Return DBCommand.ExecuteScalar()
    End Function

    Public Sub TransactionBegin()
'إنشاء مداولة جديدة لقاعدة البيانات.
        On Error Resume Next
        HoldTransaction = LibraryDB.BeginTransaction()
    End Sub

    Public Sub TransactionCommit()
'تجاهل حالة عدم وجود مداولة.
        On Error Resume Next
        If (HoldTransaction Is Nothing) Then Return

        ' ----- Commit the transaction.
```

```
'نفذ المداولة.
        HoldTransaction.Commit()
        HoldTransaction = Nothing
    End Sub

    Public Sub TransactionRollback()
        ' ----- Ignore if there is no transaction.
'تجاهل حالة عدم وجود مداولة.
        On Error Resume Next
        If (HoldTransaction Is Nothing) Then Return

        ' ----- Rollback the transaction.
'التراجع عن المداولة.
        HoldTransaction.Rollback()
        HoldTransaction = Nothing
    End Sub
```

لنشرح هذه الروتينات السبع:

دالة إنشاء جدول بيانات Function CreateDataTable

تقديم عبارة سكول ،لاستخراج نتائجها من قاعدة البيانات،ووضع جميع النتائج في كائن جدول بيانات DataTable. يتصل الكائن SqlDataAdapter إلى SqlDataReader من خلال جدول البيانات DataTable.

دالة إنشاء قارئ Function CreateReader.

تقديم عبارة سكول،لاستخراج نتائجها من قاعدة البيانات،والعودة بكائن قارئ بيانات سكول SqlDataReader المصاحب .

الإجراء تنفيذ سكول Sub ExecuteSQL .

إرسال عبارة سكول إلى قاعدة البيانات للمعالجة.

الدالة تنفيذ العودة بسكول Function ExecuteSQLReturn.

إرسال عبارة سكول إلى قاعدة البيانات للمعالجة،والعودة بقيمة لنتيجة مفردة.

إجراء بدء المداولة Sub TransactionBegin.

بدء مداولة جديدة.

إجراء تنفيذ المداولة Sub TransactionCommit .

تنفيذ المداولة.وجعل جميع التغيرات ثابتة permanent.

إجراء مسار التراجع عن المداولة Sub TransactionRollback.

السير بطريق الرجوع عن المداولة،وعدم عمل أي تغيرات والتي كانت جزء من المداولة.

ولا واحد من هذه الروتينات يتضمن كود معالجة خطأ خاص بها،فهي إما تقمع(تخمد) الأخطاء بالعبارة"عند الخطأ تابع التالي"،أو الاعتماد على الروتين المستدعي لاصطياد الأخطاء.وهذا يتيح للروتين المستدعي القيام بفعل معين بالاعتماد على نوع الخطأ الناتج.جميع هذه الروتينات متشابهة إلى حدٍ بعيد مع بعضها البعض. ففي كود الروتين CreateReader،جزء واحد مهم وهو استخدام الكائن HoldTransaction عند تكون المداولة قيد التنفيذ.

If Not (HoldTransaction Is Nothing) Then dbCommand.Transaction = HoldTransaction

معالجة قيم البيانات Processing Data Values.

يتضمن بناء عبارات سكول يدوياً الكثير من معالجة النصوص،زائد المعالجة الشرطية لها عدة مرات عند احتمال فقدان البيانات.على سبيل المثال،إذا أردت تخزين قيم نصية في قاعدة البيانات،عليك تحضيرها للاستخدام من قبل عبارة سكول(معالجة خاصة من أجل علامات الاقتباس المفردة)،ولكن إذا كانت قيمة النص ذات طول صفري zerolength، فإنك تمرر الكلمة في العبارة عوضاً عن ذلك.جميع هذه التحضيرات للبيانات يمكن أن تعرقل كودك،لذلك لما لا تكثفها(تجعلها مركزية)؟الروتينات الثمانية في هذا المقطع إما تحضر البيانات للاستخدام في عبارات سكول،أو تضبط البيانات المستخرجة للاستخدام في التطبيق.

```
Public Function DBCombo(ByRef whichField As ComboBox) As String
'تحضير قيمة صندوق مركب للاستخدام في قاعدة البيانات
        Dim listItem As Library.ListItemData
'الحصول على البيانات المخزنة لهذا البند.
        If (whichField.SelectedItem Is Nothing) Then Return "NULL"
        listItem = CType(whichField.SelectedItem, Library.ListItemData)

'(-1)كقيمة لبند ما لديها معنى خاص:غير مختارة
        If (listItem.ItemData = -1) Then
            Return "NULL"
        Else
            Return CStr(listItem.ItemData)
        End If
    End Function

    Public Function DBDate(ByVal origText As String) As String
        ' ----- Prepare a date for insertion in a SQL statement.
        If (Trim(origText) = "") Then
            Return "NULL"
```

```
        ElseIf (IsDate(origText)) Then
            Return "'" & Format(CDate(origText), "d-MMM-yyyy") & "'"
        Else
            Return "NULL"
        End If
    End Function

    Public Function DBDate(ByVal origDate As Date) As String
        ' ----- Prepare a date for insertion in a SQL statement.
        Return "'" & Format(origDate, "d-MMM-yyyy") & "'"
    End Function

    Public Function DBGetDecimal(ByRef dataField As Object) As Decimal
        ' ----- Return the decimal equivalent of an optional database field.
        If (IsDbNull(dataField) = True) Then
            Return 0@
        Else
            Return CDec(dataField)
        End If
    End Function

    Public Function DBGetInteger(ByRef dataField As Object) As Integer
        ' ----- Return the integer equivalent of an optional database field.
        If (IsDbNull(dataField) = True) Then
            Return 0
        Else
            Return CInt(dataField)
        End If
    End Function

    Public Function DBGetText(ByRef dataField As Object) As String
        ' ----- Return the text equivalent of an optional database field.
        If (IsDbNull(dataField) = True) Then
            Return ""
        Else
            Return CStr(dataField)
        End If
    End Function

    Public Function DBNum(ByVal origText As String) As String
        ' ----- Prepare a number for insertion in a SQL statement.
        If (Trim(origText) = "") Then
            Return "NULL"
        Else
            Return Trim(origText)
        End If
    End Function

    Public Function DBText(ByVal origText As String) As String
        ' ----- Prepare a string for insertion in a SQL statement.
        If (Trim(origText) = "") Then
            Return "NULL"
        Else
            Return "'" & Replace(origText, "'", "''") & "'"
        End If
    End Function
```

DBCombo

يأخذ الكود العددي numeric code المصاحب لبند مختار في أداة صندوق مركب ComboBox ويعود به كنص.إذا لم يكن هناك نص قيد الاختيار أو القيمة هي 1- يعود الروتين بـ"بدون قيمة NULL ".

DBDate(String)

تقديم نص ما يحتوي على تاريخ منسق،والعودة بتاريخ جاهز للاستخدام في عبارة سكول.

DBDate(Date)

تقديم قيمة تاريخ صحيحة،والعودة بتاريخ نصي جاهز للاستخدام في عبارة سكول.

DBGetDecimal

العودة بعدد عشري من مجموعة نتيجة،حتى ولو كان الحقل يحوي قيمة "بدون قيمة NULL ".

DBGetInteger

يعود بعدد انتغر من مجموعة نتيجة،حتى ولو كان الحقل يحوي قيمة "بدون قيمةNULL ".

DBGetText

يعود بنص من مجموعة نتيجة،حتى ولو كان الحقل يحوي قيمة "بدون قيمة".

DBNum

يحضر عدد للاستخدام في عبارة سكول .

DBText

يحضر نص للاستخدام في عبارة سكول.راجع الكود في الأعلى من أجل هذا الروتين.فالنص في عبارة سكول يجب أن يكون محاط بعلامات اقتباس مفردة،وأي علامة اقتباس لكي يتم تضمينها يجب أن تضاعف.

التركيب(الإعداد) على مستوى النظامSystem-Level Configuration

المقطع الأخير من الكود يدعم التحديث والاستخراج السريع لقيم نظام التركيب الواسع المخزنة في جدول قيم النظام SystemValueلقاعدة بيانات المكتبة.يعود الروتين GetSystemValueبالإعداد الحالي لقيمة التركيب عند التزويد بقيمة الاسمvalue name .يحدث SetSystemValue(أو يضيف،عند الحاجة)قيمة تركيب مسماة(أو محددة).كل من هذين الروتينان يظهران في الوحدة البرمجية General.

```
Public Function GetSystemValue(ByVal valueName As String) As String
        ' ----- Return the data portion of a system value name-data pair.
        Dim sqlText As String
        Dim returnValue As String
        ' ----- Retrieve the value.
        returnValue = ""
        sqlText = "SELECT ValueData FROM SystemValue WHERE UPPER(ValueName) = " & _
            DBText(UCase(valueName))
        Try
            returnValue = DBGetText(ExecuteSQLReturn(sqlText))
        Catch ex As Exception
            GeneralError("GetSystemValue", ex)
        End Try
        Return returnValue
    End Function

    Public Sub SetSystemValue(ByVal valueName As String, ByVal valueData As String)
        ' ----- Update a record in the SystemValue table.
        Dim sqlText As String
        Dim dbInfo As SqlClient.SqlDataReader
        Try
            ' ----- See if the entry already exists.
            sqlText = "SELECT COUNT(*) FROM SystemValue WHERE UPPER(ValueName) = " & _
                DBText(UCase(valueName))
            If (CInt(ExecuteSQLReturn(sqlText)) > 0) Then
                ' ----- Value already exists.
                sqlText = "UPDATE SystemValue SET ValueData = " & DBText(valueData) & _
                    " WHERE UPPER(ValueName) = " & DBText(UCase(valueName))
            Else
                ' ----- Need to create value.
                sqlText = "INSERT INTO SystemValue (ValueName, ValueData) VALUES (" & _
                    DBText(valueName) & ", " & DBText(valueData) & ")"
            End If
            ' ----- Update the value.
            ExecuteSQL(sqlText)
        Catch ex As System.Exception
            GeneralError("SetSystemValue", ex)
        End Try
    End Sub
```

الروتين GetSystemValueواضح.فهو يستخرج ببساطة قيمة مفردة من قاعدة البيانات.

على الروتين SetSystemValueأولاً اختبار فيما إذا قيمة التركيب(الإعداد configuration value) التي ستستخدم للتحديث موجود مسبقاً في قاعدة البيانات.فإذا كانت كذلك(أي موجودة)فإنه يعدل السجلات.وإلا،فإنه يضيف سجل كامل جديد.لتحديد فيما إذا كان سجل موجود أم لا،فالمطلوب إحصاء السجلات التي تطابق اسم قيمة النظامsystem value name .ويستعلم(يستفهم) هذا الروتين عن قاعدة البيانات من خلال الطريقةExecuteSqlReturn ، والتي تعود بقيمة وحيدة من استعلام query.في هذه الحالة،القيمة هي عدد السجلات المتطابقة.

```
sqlText = "SELECT COUNT(*) FROM SystemValue WHERE UPPER(ValueName) = " & _
                DBText(UCase(valueName))
```

```
If (CInt(ExecuteSQLReturn(sqlText)) > 0) Then
```

كما ترى استخدام الروتين GetSystemValue سهل،لذلك لنستخدمه الآن.ارجع إلى معالج حدث MyApplication_Startup في *ApplicationEvents.vb* وأضف التالي إلى نهاية معالج الحدث:

```
' ----- Check the database version.
            Dim productionDBVersion As String
            productionDBVersion = Trim(GetSystemValue("DatabaseVersion"))
            If (CInt(Val(productionDBVersion)) <> UseDBVersion) Then
                MsgBox("The program cannot continue due to an incompatible database. " & _
                    "The current database version is '" & productionDBVersion & _
                    "'. The application version is '" & UseDBVersion & "'.", _
                    MsgBoxStyle.OkOnly Or MsgBoxStyle.Critical, ProgramTitle)
                CleanUpProgram()
                Me.HideSplashScreen()
                e.Cancel = True
                Return
            End If
```

بين فترة وأخرى،أجد من الضروري تعديل تركيب قاعدة البيانات إلى حد أن الإصدارات القديمة من التطبيق إما لم تعد تعمل،أو أن تسبب البيانات الرئيسية وجع راس. ولمنع هذا،عملت على إضافة إعداد إصدار قاعدة البيانات DatabaseVersion، واستخدام مقطع هذا الكود من أجل الاختبار عليه.فإذا لم يوافق البرنامج إصدار قاعدة البيانات المتوقع،فإنه سيرفض التشغيل.

الأمن Security

الأسرار أشياء ممتعة،فمع البلايين من الناس على الكوكب،لايوجد قحط في الأحداث والقصص الحقيقية،ولكن ولا واحدة منها تحفظ متعتنا إذا لم يكن هناك سر يستكشف في مكان ما.العديد من الكتب عملت على تضمين الكلمة "سر *Secret* " في عناوينها لجعلها وجعل مواضيعها أكثر إثارةً،عناوين مثل "أسرار الطبخ الياباني*Japanese Cooking Secrets* "

في هذه الفترة era تكاثف المعلومات information overload والمعايير الأخلاقية المتساهلة بإفراط increasingly permissive moral standards على التلفاز،تبدو الأسرار نادرة.ولكن كل واحد لديه معلومات مهمة يحتاج حفظها عن الآخرين،وهذا يتضمن مستخدمي برامجك.
لحسن الحظ،برامج الدوت نت والبيانات المتعلقة بها يمكن أن تكون آمنة عند الحاجة،إذا استخدمت ميزات الأمان المتاحة لك في إطار عمل الدوت نت.

ميزات الأمن في الدوت نت Security Features in .NET

يتضمن الأمان في الدوت نت العديد من الميزات،ولكنها تقع بشكل عام تحت ثلاث مناطق رئيسية:

الأمن الداخلي *Internal security*

الفئات Classesوأعضاء الفئات في الدوت نت يمكن أن يتم حمايتها بواسطة الأمن المعتمد على المستخدم أو الأمن المعتمد على القواعد. ويتواجد أمن الوصول للكود (CAS) *Code Access Security* لحفظ المستخدمين الغير مرخص بهم unauthorized من الوصول لميزات المكتبات الفعالة powerful libraries التابعة للدوت نت.وفقط هؤلاء المستخدمين الذين لديهم مجموعة قليلة أو خاصة من الحقوق يمكنهم استخدام هذه الميزات المحمية.

الأمن الخارجي *External security*

بما أن أي واحد يمكنه أن يطور وينشر تطبيق دوت نت،فمن الهام حماية مصادر النظام من كود مؤذٍ.وهذه قضية كبيرة،وخاصةً مع التقارير الذاهبة(الخارجة) من الهكرز(القراصنة) والتي تستفيد من مشاكل"اجتياح الجدار" في البرمجيات الصادرة من ميكروسوفت والبائعين الآخرين.وكما أن "أمن الوصول للكود(CAS) *Code Access Security* "يحفظ الكود من الوصول لميزات خاصة بالفئة،وهذا الأمن يتفاعل مع نظام التشغيل لحفظ الكود المؤذي(المحتال)من الوصول لبعض أو كل الملفات والأدلة،مدخلات التسجيل،مصادر شبكة العمل،محيطات أجهزة الكمبيوتر المركبة عليه،أو مجمعات الدوت نت الأخرى المعتمدة على سياسات الأمن السارية(النافذة المفعول.in-effect security policies).

أمن البيانات *Data security*

البرامج ومصادر الكمبيوتر ليست الشيء الوحيد الذي يحتاج الحماية.بعض المستخدمين ذوي الألفاظ الطنانة يفكرون أن بياناتهم ثمينة وهامة بحيث تستحق أن يتم حمايتها من خلال وسائل برمجية خاصة.التشفير Encryption والتواقيع الرقمية digital signatures وميزات تشفير cryptographic أخرى توفر دعم خاص تحتاجه مثل هذه البيانات.
ولأن مشروع المكتبة يتفاعل مع مصادر خارجية رئيسية – قاعدة بيانات سكول سرفر- فهي تتعامل مع قضايا الأمن الخارجي،وأيضاً بطريقة غير مباشرة مع ADO.NETوسياسات أمن الأنظمة system security policies.ولأن الكتاب يركز على توزيع تطبيق عمل نموذجي ،فهذا الفصل لن يناقش كل من قضايا الأمن الخارجي والداخلي،ولكنه سيركز على مواضيع أمن البيانات،وخاصة تشفير البيانات.

التشفير وعلم التشفير Cryptography and Encryption

استيداع سر،لدى شخص أخر هي قضية،والتأكد من أن سر ما يأتي من شخص أخر موثوق هي قضية أخرى أيضاً.التأكد من أنني أحصل على أفضل معاملة بالنسبة لتأمين السيارة هي قضية أخرى مختلفة بالكامل.
من الواضح،أمن البيانات هي أكثر من حفظ قطعة من البيانات محمية عن أعين المتطفلين.وليس ما يهمنا فقط أعين المتطفلين.

حفظ الأسرار Keeping Secrets

عندما يفكر الناس بشأن التشفير encryptionوأمن البيانات data security،بشكل عام يركزون على سمة"حفظ الأسرارkeeping secrets".إمكانية كتابة محتوى بشكل كودي(تحويل الأحرف إلى رموز)،وحفظها بعيدة عن الخصمadversary،مع ذلك تبقى قابلة لفك التشفير من قبلك أو إرفاقها في وقت متأخر هو هام.مجال تقنيات التشفير بدءاً من الانحرافات عن اللغة البسيطة وتبديل الشفرة(استبدال الأحرفletter substitutions ،المستخدمة في الألغاز الرمزية(المكتوبة بالرموزcryptogram puzzles))إلى أنظمة التكويد النوعية للأجهزة التفسيرية enigma-machine-quality encoding systems.البرمجيات التي تمكن التشفير هي جزء من تعلمنا اليومي الآن.عندما تعمل كرت اعتماد (بطاقة ائتمان)شرائية من مواقع انترنت،فإن الفرصة جيدة نسبياً لأن تكون معلومات بطاقة الاعتماد خاصتك يتم تشفيرها وتحويلها إلى دقة سرية في 128- بت.
نموذجياً طرق التشفير encryptionتعمل على استخدام واحد أو أكثر من المفاتيح،زائد توحيد دوال العنونة بالكود*hashing functions*وخوارزميات التشفير*encryption algorithms*، لتحويل محتوى دقيق sensitive contentإلى صيغة ليس من السهل الوصول لها دون المفتاح المناسب أو الأصلي.التكويد المتماثل Symmetric cryptography هو الاسم المستخدم لطرق التشفير encryption التي تستخدم مفتاح سري مفرد single secret key.يعرف أيضاً التشفير بالمفاتيح العامة *Public-key encryption* بالتشفير الغير متماثل *asymmetric cryptography* -ويستخدم زوج من المفاتيح لتشفير encryptوفك تشفير decryptالبيانات.يمكن أن يتم إعطاء المفاتيح العامة لأي واحد يهتم بالاتصال معك بشكل سري.حتى من الممكن أن تعطي هذه المفاتيح لمنافسيك،فهي عامة.
المفاتيح الخاصة *private key* المناسبة يتم حفظها بشكل آمن لتستخدمها أنت،ولن تريها لأي كان.المحتوى المشفر باستخدام واحد من المفاتيح(وخوارزمية التشفير)يمكن أن يتم فك تشفيره فيما بعد باستخدام المفتاح الآخر.إذا ما شفر صديقك بعض المعلومات باستخدام مفتاح عام،لايمكن لأي أن يكون قادر على فكه ما عداك أنت،وسيتطلب مفتاح خاص بك.تستطيع أيضاً تشفير البيانات بواسطة مفتاحك الخاص،ولكن أياً كان يستطيع فك شفرته بواسطة المفتاح العام.سنرى استخدام هذا الإجراء الغير آمن ظاهرياً بعد قليل.

استقرار البيانات Data Stability

يساعد تشفير البيانات ضمان تكامل مقطع من البيانات،حتى ولو كانت تلك البيانات غير مشفرة.إذا أرسلت لشخص أيميل أثناء عاصفة رعدية،توجد وبشكل خاص فرصة لأن يتم تبديل بعض أو كل محتوى الرسالة الالكترونية بشكل الكتروني قبل وصولها إلى المستقبل.دعنا نقول أن بعض الكهرباء الساكنة في خطوط شبكة النقل حدث لأن تسبب مضاعفة جملة واحدة .كيف ستعرف فيما إذا كان المؤلف يحاول صنع نقطة تذكير أو ببساطة شائبة كمبيوتر؟.

تضمين "اختبار الإضافة *checksum* " مع المحتوى يساعد في التعرف على مشاكل البيانات أثناء النقل.يدعى "اختبار الإضافة *checksum* "في بعض الأحيان "قيمة التكرار *hash value* "- تأخذ المحتوى الأصلي وتمرره إلى دالة تولد قيمة مختصرة short value تمثل البيانات الأصلية.دوال "اختبار الإضافة"(أو خوارزميات التكرار*hashing algorithms*)حساسة جداً حتى للتغيرات في البايت المفرد ضمن المحتوى،فيما إذا تم تبديل ذلك البايت المفرد،تغير موقعه،أضيف،أو تمت إزالته من البيانات الأصلية.بتوليد"اختبار الإضافة"قبل وبعد إرسال البيانات،تستطيع التثبت فيما إذا تغير المحتوى بالكامل أثناء النقل.

تمثل "اختبارات الإضافة" تشفير غير موجه للبيانات الأصلية.من غير الممكن استخدام "اختبار الإضافة" للحصول على محتوى البيانات الأصلية.وهذا صحيح،إذاً،وبما أن الهدف من "اختبار الإضافة" ليس لتسليم المحتوى بشكل سري،ولكن تسليمها غير متغيرة.التشفير الثنائي الاتجاه *Bidirectional encryption* هو ما تحدثت عنه في مقطع"حفظ الأسرارKeeping Secrets ".إذا كان لديك مفتاح صحيح و خوارزمية صحيحة،يعيد التشفير الثنائي الاتجاه bidirectional encryption تخزين المحتوى الأصلي من المحتوى المشفر.

إثبات الهوية Identity Verification

لنقول أنك استقبلت رسالة الكترونية تقول شيء ما،كيف تعرف أن هذه الرسالة موثوقة،أو حقيقة من مرسل معين؟في هذه الحالة،المحتوى لوحده يجب أن يبرهن استحقاقه الثقةtrustworthy .ولكن إذا أردت حقاً التحقق من المصدر،ومن غير المتاح الاتصال مع من بعث الرسالة،تستطيع توظيف تواقيع رقمية *digital signatures* لإثبات هوية المرسل.

طريقة وحيدة لاستخدام التواقيع الرقمية التي توظف تشفير بالمفاتيح العامة لنقل رسالة أو كلمة مرور متفق عليها،وتمرر المحتوى المشفر على طول مع رسالة الكترونية أكبر.على سبيل المثال،يمكن أن يشفر المرسل النص"أنا المرسل فلان "باستخدام مفتاحه الخاص .عندما تستقبل الايميل،يمكنك فك تشفير التوقيع الرقمي باستخدام مفتاح المرسل العام.إذا أنتج فك تشفير الرسالة"أنا المرسل فلان "،ستعلم أن تلك الرسالة فعلية،وفي الحقيقة تأتي من المرسل فلان.

التشفير في الدوت نت.Encryption in .NET

تشفير البيانات وميزات الأمن المضمنة مع الدوت نت تظهر في فضاء الأسماءSystem.Security.Cryptography .معظم الفئات في فضاء الأسماء هذا تنفذ خوارزميات تشفير معروفة جيداً ومتنوعة والتي تم قبولها بواسطة المنظمات والحكومات كمعايير تشفير معتمدة.على سبيل المثال الفئة DESCryptoServiceProviderتوفر ميزات معتمدة على خوارزمية معيار تشفير البيانات Data Encryption Standard (DES) algorithm،الخوارزمية المطورة أصلاً بواسطة شركة IBMفي منتصف1970.

علم التكويد(التشفير) المتماثلSymmetric Cryptography

يستخدم علم التكويد المتماثل مفتاح سري مفرد single secret key لتشفير وفك تشفير مقطع بيانات.على الرغم من أن هذه الخوارزميات وعلى الأغلب سريعة جداً (عند مقارنتها مع التكويد غير المتماثلasymmetric cryptography)الحاجة لتوفير مفتاح سري كامل للآخرين لمشاركة البيانات يمكن أن تجعلها(البيانات) ملازمة لأمن أقل.مايزال كافي بالنسبة لعدة تطبيقات،"التشفير بمفتاح سريsecret key encryption ".

يتضمن إطار عمل الدوت نت دعم لأربع خوارزميات تشفير متماثل:

- معيار تشفير البيانات(*DES*) *Data Encryption Standard* ، شفرة مقطع block cipher 56-بت مع دعم رئيسي من خلال الفئة DESCryptoServiceProvider.هذه الخوارزمية بشكل عام آمنة،ولكن تبعاً لحجم مفتاحها الصغير(المفاتيح الأصغر أكثر سهولة في الفضح)،وهي غير مناسبة من أجل البيانات العالية الحساسية.

- شفرة ريفست رقم2(*Rivest Cipher number 2*) *RC2*،شفرة مقطع 56- بت مع دعم رئيسي من خلال الفئة RC2CryptoServiceProvider.

- ريجدل *Rijndael* (مشتقة من اسمي مصمميها،ديمن Daemenوريجمن Rijmen) شفرة مقطعية ذات بت متغير(بين 128 إلى 256 بت) block cipher مع دعم رئيسي من خلال الفئة RijndaelManaged.ويتم تعليقها إلى خوارزمية مشابهة مسماة"معيار تشفير متقدم Advanced Encryption Standard (AES) "،وهي الخوارزمية الأكثر أمناً من خوارزميات مفتاح السر المزودة بواسطة الدوت نت.

- معيار تشفير البيانات الثلاثي *Triple DES*،شفرة مقطعية تستخدم الخوارزمية DESalgorithmالمضمنة ثلاث مرات لتوليد نتيجة أكثر أمناً،مع دعم رئيسي من خلال الفئةTripleDESCryptoServiceProvider .على الرغم من أنها أكثر أمناً من DES البسيطة،فما تزال أقل تحصيناً vulnerableمن ريجدل Rijndaelأو القياسيAES .

فئات "المزود provider "المتنوعة هي أدوات يجب أن يتم استخدامها مع بعضها ومع فئات التكويد لتعمل بشكل مناسب.على سبيل المثال،هذه العينة من الكود(المعتمدة على كود موجود في توثيق MSDN)يستخدم الفئة DESCryptoServiceProviderوالفئة CryptoStream،كلاهما أعضاء من System.Security.Cryptography، معاً لتشفير وفك تشفير مقطع نصي :

```
Imports System
```

```
Imports System.IO
Imports System.Text
Imports System.Security.Cryptography
Class CryptoMemoryStream
    Public Shared Sub Main()
        ' ----- Encrypt then decrypt some text.
        Dim key As New DESCryptoServiceProvider
        Dim encryptedVersion() As Byte
        Dim decryptedVersion As String
        ' ----- First, encrypt some text.
        encryptedVersion = Encrypt("This is a secret.", key)
        ' ----- Then, decrypt it to get the original.
        decryptedVersion = Decrypt(encryptedVersion, key)
    End Sub
    Public Shared Function Encrypt(ByVal origText As String, _
    ByVal key As SymmetricAlgorithm) As Byte()
        ' ----- Uses a crytographic memory stream and a
        ' secret key provider (DES in this case)
        ' to encrypt some text.
        Dim baseStream As New MemoryStream
        Dim secretStream As CryptoStream
        Dim streamOut As StreamWriter
        Dim encryptedText() As Byte
        ' ----- A memory stream just shuffles data from
        ' end to end. Adding a CryptoStream to it
        ' will encrypt the data as it moves through
        ' the stream.
        secretStream = New CryptoStream(baseStream, _
        key.CreateEncryptor(), CryptoStreamMode.Write)
        streamOut = New StreamWriter(secretStream)
        streamOut.WriteLine(origText)
        streamOut.Close()
        secretStream.Close()
        ' ----- Move the encrypted content into a useful
        ' byte array.
        encryptedText = baseStream.ToArray()
        baseStream.Close()
        Return encryptedText
    End Function
    Public Shared Function Decrypt(ByVal encryptedText() As Byte, _
    ByVal key As SymmetricAlgorithm) As String
        ' ----- Clearly, this is the opposite of the
        ' Encrypt( ) function, using a stream reader
        ' instead of a writer, and the key's
        ' "decryptor" instead of its "encryptor."
        Dim baseStream As MemoryStream
        Dim secretStream As CryptoStream
        Dim streamIn As StreamReader
        Dim origText As String
        ' ----- Build a stream that automatically decrypts
        ' as data is passed through it.
        baseStream = New MemoryStream(encryptedText)
        secretStream = New CryptoStream(baseStream, _
        key.CreateDecryptor(), CryptoStreamMode.Read)
        streamIn = New StreamReader(secretStream)
        ' ----- Move the decrypted content back to a string.
        origText = streamIn.ReadLine()
        streamIn.Close()
        secretStream.Close()
        baseStream.Close()
        Return origText
    End Function
End Class
```

يضم هذا الكود فئة DESencryptionمع *stream* ،أداة مشتركة في تطبيقات الدوت نت لنقل البيانات من حالة أو موقع لأخر.("الجداول(أو التجميعات) Streams "هي طريقة رئيسية مستخدمة لقراءة وكتابة الملفاتfiles)الجداول Streamsليست بأداة صعبة الاستخدام،ولكن ماتزال تبدوا معقدة convoluted قليلاً. لماذا لا تتضمن الفئة DESCryptoServiceProvider طرق تشفير Encryptوفك التشفير Decrypt ببساطة ؟هذا سؤالي،على

الأقل،إني متأكد من أن لديها شيء ما يجب عمله مع حفظ الفئة "شاملة generic "للاستخدام في بيئات بيانات متعددة.مايزال،هذا الكود كثيف ،من المؤكد أنه أكثر سهولة من كتابة كود التشفير بنفسي.وهو عام بشكل كافي بحيث أستطيع مبادلة خوارزميات مفتاح سري دون تغير كبير في الكود.

التكويد غير المتماثل Asymmetric Cryptography

في تكويد المفتاح السري،تستطيع استخدام أي مفتاح قديم تريده لدعم عملية التشفير وفك التشفير.طالما أنك تحفظه سراً، في الحقيقة محتوى المفتاح نفسه ليس مهم كثيراً.لايمكن قول المثل على التكويد الغير متماثل(المفتاح العام).لأن المفاتيح المفصولة يتم استخدامها لتشفير وفك تشفير البيانات،المفاتيح الخاصة والعامة النوعية يجب أن يتم صنعها بشكل خاص كــزوج.لاتستطيع اختيار فقط مفاتيح خاصة وعامة عشوائية وتأمل أن تعمل مع بعضها.

المكونات المستخدمة لدعم التكويد الغير متماثل تتضمن"المولداتgenerators" والتي تصدر emitأزواج مفاتيح خاصة وعامة.حالما يتم التوليد،هذه المفاتيح يمكن أن يتم استخدامها في كودك لتقنيع maskالبيانات الحساسة.وتبعاً لحجم المفتاح الأكبر،فمن الصعب بالنسبة لكل واحد أن يدخل تعديل على بياناتك المشفرة.

تشفير المفتاح العام مشهور notoriouslyببطئه،فهو يأخذ دائماً يوم لكتابة كود كميات ضخمة من البيانات باستخدام المفتاح المصدري.بسبب بطء sluggishأداء التشفير الغير المتماثل،فالعديد من أنظمة أمن البيانات تستخدم جمع من تشفير مفتاح عام ومفتاح سري لحماية البيانات. التراخيص الأولية تحدث مع عمليات المفتاح العام،ولكن حالما تفتح قناة الأمن،البيانات الممررة بين الأنظمة التي تحصل على التشفير باستخدام طرق المفتاح السري الأسرع.تتضمن الدوت نت فئتي تكويد تفي برغبتك في التشفير وفك التشفير:

.خوارزمية التوقيع الرقمي(*DSA*) *Digital Signature Algorithm*، خوارزمية صممت بواسطة الحكومة الأمريكية لاستخدام التواقيع الرقمية،مع دعم رئيسي من خلال الفئة DSACryptoServiceProvider.

.خوارزمية *RSA*(موجديها Ron Rivest و Adi Shamir،و Adleman)،خوارزمية غير متماثلة الأمن أقدم مع دعم رئيسي من خلال الفئة RSACryptoServiceProvider.

سوف نستخدم التشفير الغير متماثل في مشروع المكتبة،ولكن ليس حتى أخر فصل.حتى ذلك الحين،لن أدخل بالكثير من التفاصيل حول أعمال التشفير الغير متماثل.

البعثرة(أو الخلط) Hashing

على الرغم من أن خوارزميات البعثرة(خوارزميات العنونة بالكود(أو الرموز)) hashing algorithms لا تمنحك إمكانية تشفير وفك تشفير البيانات كما ترغب،ولكنه مفيدة في دعم الأنظمة التي تؤمن وتتحقق من محتوى البيانات.سننجز بعض البعثرة على البيانات في كود المشروع من أجل هذا الفصل،لذلك ابق يقظ .

إيجاد حل خوارزمية البعثرة سهل.فهي أخذت أفضل عقول وكالة الأمن العالمية ومعهد ولاية ماسشتس للتكنولوجيا لاختراع أنظمة التشفير بمفتاح عام ومفتاح خاص،ولكنك تستطيع تطوير خوارزمية البعثرة بعدة دقائق فقط إليك خوارزمية بسيطة قمت بعملها :

```
Public Function HashSomeText(ByVal origText As String) As Long
        ' ----- Create a hash value from some data.
        Dim hashValue As Long = 0&
        Dim counter As Long
        For counter = 1 To Len(origText)
            hashValue += Asc(Mid(origText, counter, 1))
            If (hashValue > (Long.MaxValue * 0.9)) Then _
            hashValue /= 2
        Next counter
        Return hashValue
    End Function
```

في الكود عملت فقط على جمع قيم الآسكي لكل حرف في السلسلة الحرفية للنص،وعملت على إعادة النتيجة.وقد عملت اختبار في الحلقة للتأكد من أنني لا أزيد عن 90% من قيمة الطول الأعظمي،ولا أريد تجاوز المتغير hashValueوتوليد خطأ.على الرغم من أن الدالة HashSomeTextتعمل على توليد تمثيل مبعثر hashed representation للبيانات المدخلة، لديها أيضاً بعض العيوبdeficiencies :

.من السهل جداً التخمين من قيمة التكرار hashفيما إذا المحتوى القادم كان قصير أو طويل.المحتوى الأقصر سيولد عدد صغير،وقيم المخرجات الأكبر تميل إلى الإشارة إلى محتوى إدخالات أكبر.

.ليست ذات حساسية عالية لبعض أنواع تغيرات المحتوى.على سبيل المثال،إذا رتبت عدة حروف في المحتوى،من المحتمل أن لا تؤثر على قيمة البعثرة.تغيير الحرف سيؤثر على القيمة ولكن إذا غيرت حرف من *A* إلى *B*وحرف أخر مجاور من *T* إلى *S* فإن قيمة البعثرة ستبقى نفسها ولن تتغير.

.كلما قصر المحتوى كلما زادت فرص توليد نفس قيمة البعثرة بالنسبة لإدخالين.

من المحتمل أنك تحتاج شيء ما أقوى.إذا كان الأمر كذلك،تتضمن الدوت نت العديد من أدوات البعثرة:

1.بعثرة معتمدة على كود ترخيص(توثيق)الرسالة (HMAC) *Hash-based Message Authentication Code*يتم حسابه باستخدام دالة بعثرة خوارزمية أمن البعثرة رقم1(SHA-1) Secure Hash Algorithm number 1، يمكن جعلها متاحة من خلال الفئة HMACSHA1.وتستخدم كود بعثرة 160 بت. لايوجد قيود restrictions خاصة على طول مفتاح السر secret keyالمستخدم في الحساب.على الرغم من أنها مناسبة لحالات الخطر المنخفضة،فإن الخوارزمية SHA-1عرضة للهجوم(الغزو).

2.كود توثيق الرسالة(MAC) *Message Authentication Code* يتم حسابه باستخدام خوارزمية المفتاحTriple-DESsecret (تم شرحها سابقاً)،يمكن جعلها متاحة من خلال الفئةMACTripleDES. مفتاح السر secret key المستخدم في الحساب هو إما بطول 16 أو 24بايت،والقيمة المولدة بطول 8بايت.

3.حساب البعثرة لـخوارزمية استيعاب الرسالة رقم5(MD5) *Message-Digest algorithm number 5*، يمكن جعلها متاحة من خلال الفئة MD5CryptoServiceProvider.إن MD5ما تزال خوارزمية سرية عالية المستوى صممت من قبل رون ريفيستRon Rivest، ولكنها أظهرت احتواءها على بعض العيوب التي تجعل من أمن التشفير خطرencoding security risk. قيمة التكرار الناتجة هي بطول128بت.

4.مثل الفئة HMACSHA1،تحسب الفئة SHA1المدارة SHA1Managed class قيمة البعثرة باستخدام دالة البعثرةSHA-1. مهما تكن،تم كتابتها باستخدام كود الدوت نت المدار NET managed code. فقط.إن HMACSHA1وبعض ميزات التكويد الأخرى في الدوت نت بكل بساطة تدور حول (تغلف)(CAPI) *Cryptography API* الأقدم،ومكتبات الربط الديناميكي للدوت نت السابقة NET DLL.-SHA1.pre المدارة تستخدم كود بعثرة 160بت.

5.فئات ثلاث أخرى:SHA256 المدارة، SHA384المدارة، SHA512المدارة، مشابهة لفئة SHA1المدارة،ولكن تستخدم أكواد بعثرة 256بت ،384بت،و512بت على الترتيب.

كل من هذه الخوارزميات تستخدم مفتاح السر الذي يجب أن يتم تضمينه كل مرة يتم توليد البعثرة على نفس المجموعة من البيانات المدخلة.طالما أن البيانات المدخلة لم تتغير ،ومفتاح السر هو نفسه،فقيمة التكرار الناتج ستبقى غير متغيرة.تصميمياً،حتى أصغر تغير في البيانات المدخلة يولد تغيرات رئيسية في قيمة البعثرة المخرجة.

ميزات أمن أخرىOther Security Features

يوجد العديد من ميزات الأمن الهامة والتي لن أناقشها هنا بالتفصيل،ولكن تستحق على الأقل أن أشير إليها.

ترخيص المستخدم و (My.User). User Authentication and My.User

يتضمن كائن الفيجوال بيسك My.User العديد من ميزات الترخيص(التوثيق)authentication) والتي يمكن أن تساعد على تصميم كود تمكين الأمن.عضو واحد مفيد هو خاصية "الاسم Name "،والتي تزود باسم المستخدم المرخص حالياً.تخبرك الطريقة IsInRole فيما إذا المستخدم الفعال يتم تضمينه ،فيما يدعى مجموعة أمن المدراءthe Administrators security group. من أجل تطبيقات نماذج ويندوز،ستشير أعضاء My.User بشكل نموذجي إلى تسجيل دخول مستخدم ويندوز. مهما يكن ،تستطيع استخدام أنظمة توثيق authentication systems أخرى والتي تلبي احتياجات التطوير الخاصة بك.تتضمن الخيارات Optionsاستخدام الانترنت المعتمد على نظام "معرف ويندوز لايف" Internet-based “Windows Live ID” من ميكروسوفت،وأنظمة توثيق ثانوية مشاركة أخرىthird-party authentication systems ،أو نظام إدارة المستخدم المخصص التصميم من قبلك.

فئة أمن النصوص The SecureString Class

إنه لمن المميز فمع كل هذه الأدوات المتقدمة ،ما يزال المستخدمين يمضون الكثير من الوقت في البناء وتفسير البيانات النصية.لحسن الحظ،تتضمن الدوت نت وفرة من أدوات مفيدة لمعالجة النصوص .ولكن لسوء الحظ،هي ليست آمنة كثيراً.من المحتمل أنك تتذكر أن نصوص الدوت نت ثابتة immutable،حالما يتم إنشاءها،فهي لن تتغير.أخيراً،سيتم تدميرها بواسطة معالجة مجمع النفايات.ولكن حتى ذلك الحين،فإنها تستقر في الذاكرة،منتظرةً أن يتم عمل مسح بواسطة كود المصمم من قبل هكر ما.بشكل داخلي،البيانات النصية يتم تخزينها كنصوص سهلة،لذلك إذا ما استطاع شخص ما الحصول على الذاكرة،يستطيع نسخ المحتوى لأهداف شريرة nefarious .

تتيح لك الفئة System.Security.SecureStringتخزين النصوص والحصول عليها مرةً أخرى،ولكن داخلياً،محتوى النص يتم تشفيره.إذا ما حصل شخص ما على محتوى داخلي لحالة فئة،سيبدو مثل الثرثرةgibberish .

مشروع Project

هذا الفصل سيبين ميزات متمحورة حول مواضيع الأمن التالية والمضافة لمشروع المكتبة:

. فورم تسجيل الدخول" login "،والتي توثق(تثبت صحة)أمناء المكتبة،والمستخدمين الإداريين الآخرين.

.نماذج مجموعة الأمن وإدارة المستخدم.

.دالة functionتعمل على تشفير كلمة المرور المزودة من قبل المستخدمuser-supplied password .

.تفعيل بعض ميزات التطبيق والتي تعتمد على توثيق(تصديق)authentication) المستخدم.

دعم التصديق(الترخيص) Authentication Support

بما أن جميع بيانات المكتبة يتم تخزينها في قاعدة بيانات سكول سرفر،فقد استخدمنا سابقاً إما ويندوز Windowsأو أمن مخدم سكول SQL Server security (أمن سكول سرفر) لتقيد restrict الوصول إلى البيانات نفسها.ولكن حالما نتصل بقاعدة البيانات،سنستخدم نظام تصديق(توثيق)مخصص لتمكين وعدم تمكين ميزات في التطبيق.فهو موجود هناك بحيث نضع بعض ميزات تشفير دوت نت قيد الاستخدام.

قبل إضافة الكود المهم،نحتاج لإضافة بعض المتغيرات العامة والتي تدعم الأمن على كامل التطبيق.جميع هذه العناصر العامة تظهر في الملف *General.vb*، ضمن كود الوحدة البرمجية .

```
Public LoggedInUserID As Integer
    Public LoggedInUserName As String
    Public LoggedInGroupID As Integer
```

```
    Public SecurityProfile(MaxLibrarySecurity) As Boolean
```

على الرغم من أننا عملنا على إضافة العداد LibrarySecurityسابقاً،فهو جزء هام من نظام الأمن.وعناصره تطابق ما يتواجد في جدول Activity في قاعدة بيانات المكتبة.كل قيمة للعداد توافق عنصر في المصفوفة SecurityProfileوالتي أضفناها الآن إلى الكود.

```
'قيم الأمن
    Public Enum LibrarySecurity As Integer
        ManageAuthors = 1
        ManageAuthorTypes = 2
        ManageCopyStatus = 3
        ManageMediaTypes = 4
        ManageSeries = 5
        ManageGroups = 6
        ManageItems = 7
        ManagePatrons = 8
        ManagePublishers = 9
        ManageValues = 10
        ManageUsers = 11
        ProcessFees = 12
        ManageLocations = 13
        CheckOutItems = 14
        CheckInItems = 15
        AdminFeatures = 16
        DailyProcessing = 17
        RunReports = 18
        PatronOverride = 19
        ManageBarcodeTemplates = 20
        ManageHolidays = 21
        ManagePatronGroups = 22
        ViewAdminPatronMessages = 23
    End Enum
    Public Const MaxLibrarySecurity As LibrarySecurity =LibrarySecurity.ViewAdminPatronMessages
```

جميع المتغيرات العامة الجديدة المضافة تخزن معلومات مطابقة للمدير administrator الفعال.عندما يكون زبون هو المستخدم الفعال،يضع البرنامج هذه القيم لإعداداتها الافتراضية.بما أن هذا يجب عمله عندما يبدأ البرنامج للمرة الأولى،سنضيف روتين InitializeSystem يتم استدعاءه عند البدء.ويظهر هذا الروتين أيضاً في الوحدة البرمجية General.

```
Public Sub InitializeSystem()
        ' التمهيد للمتغيرات العامة هنا.
        Dim counter As Integer
        ' إزالة قيم الأمن ذات الصلة.
        LoggedInUserID = -1
        LoggedInUserName = ""
        LoggedInGroupID = -1
        For counter = 1 To MaxLibrarySecurity
            SecurityProfile(counter) = False
        Next counter
    End Sub
```

(لدى مصفوفة SecurityProfileبنود تتراوح من 0إلى MaxLibrarySecurity،ولكن الحلقة عند نهاية هذا الكود تبدأ من العنصر1،لأن جدول Activity يبدأ العد من 1 ،فقررت تجاوز العنصر0).يتم استدعاء الطريقة InitializeSystem من الحدث MyApplication_Startup في ملف *ApplicationEvents*.vb، تماماً قبل تأسيس الاتصال إلى قاعدة البيانات.لذلك دعنا نضيف الكود الآن.

```
'عمل تمهيد عام
            InitializeSystem()
```

كل مرة يحاول مدير استخدام النظام،وكل مرة يسجل خروج ويعود البرنامج إلى نمط الزبون(العميل)،فإن جميع المتغيرات العامة ذات الصلة يجب أن يتم إعادة وضعها.وهذا يتم عمله في الطريقةReprocessSecuritySet ،أضف هذه الطريقة إلى الوحدة البرمجية General.

```
Public Sub ReprocessSecuritySet()
        ' إعادة التحميل في مجموعة الأمن للمستخدم الحالي.
        ' لو لم يكن مستخدم مسجل الدخول,نظف جميع الإعدادات
        Dim counter As Integer
        Dim sqlText As String
        Dim dbInfo As SqlClient.SqlDataReader = Nothing
        ' أزل البنود الموجودة
        For counter = 1 To MaxLibrarySecurity
            SecurityProfile(counter) = False
        Next counter
        ' أخرج إذا لم يكن هناك مستخدم قد سجل الدخول.
        If (LoggedInUserID = -1) Or (LoggedInGroupID = -1) Then Return
        Try
```

```
'                                    التحميل في عناصر الأمن لهذا المستخدم
            sqlText = "SELECT ActivityID FROM GroupActivity WHERE GroupID = " & LoggedInGroupID
            dbInfo = CreateReader(sqlText)
            Do While (dbInfo.Read)
                SecurityProfile(CInt(dbInfo!ActivityID)) = True
            Loop
            dbInfo.Close()
        Catch ex As Exception
'                                    بعض الأخطاء المتعلقة بقاعدة البيانات
            GeneralError("ReprocessSecuritySet", ex)
            If (dbInfo IsNot Nothing) Then dbInfo.Close()
'                    عدم دخول المدير يؤدي إلى اللجوء إلى التكرار المستمر
            LoggedInUserID = -1
            LoggedInGroupID = -1
            ReprocessSecuritySet()
        Finally
            dbInfo = Nothing
        End Try
    End Sub
```

يستخدم هذا الروتين كود تم بناءه في الفصل العاشر وفصول سابقة.فعندما يكتشف هذا الروتين مستخدم موثوق(مصرح به)(أي المتغير LoggedInUserID)فإنه ينشئ الكائن SqlDataReader مع ميزات الأمن المسموحة للمستخدم،ويخزن تلك الإعدادات في المصفوفة SecurityProfile.حالما يتم تحميلها،فيمثل أي عنصر للمصفوفة عندما يكون صحيح True ميزة التطبيق أن المدير administrator مصرح به كي يستخدم التطبيق.سأناقش جدول GroupActivity بعد قليل في هذا الفصل.

إذا ما حدث خطأ قاعدة بيانات خلال المعالجة،يعمل الكود على إعادة وضع كل شيء إلى نمط العميل،وعمل استدعاء دوري *recursive* لـReprocessSecuritySet لتنظيف المصفوفة SecurityProfile. (يحصل التكرار المستمر Recursion عندما يستدعي روتين نفسه بشكل مباشر أو غير مباشر.)

تشفير كلمات المرور Encrypting Passwords

سأستخدم واحدة من طرق خلط(بعثرة) الدوت نت لتشفير كلمة المرور المزودة من قبل المدير قبل تخزينها في قاعدة البيانات.واحد من جداول قاعدة بيانات المكتبة،جدول UserName،يخزن قالب profile الأمن الأساسي لكل مكتبي أو مستخدم إداري أخر،ومن ضمنها كلمات المرور.بما أن أياً كان يستطيع الدخول إلى قاعدة البيانات سنكون قادرين على رؤية كلمات المرور المخزنة في هذا الجدول،فسنعمل على تشفيرها لجعلها أقل جذباً.(من أجل الزبائن الذين يستخدمون البرنامج ببساطة،فليس من الواجب عليهم الوصول مباشرةً إلى قاعدة البيانات بعيداً عن التطبيق،ولكنك لاتعلم هؤلاء الزبائن اللعوبين).

لحفظ الأشياء آمنة،فسنشوش(نمزج scramble)كلمة المرور المدخلة من قبل المستخدم،واستخدامها لتوليد قيمة خلط(بعثرة)وتخزين قيم البعثرة في حقل كلمة المرور password لقاعدة البيانات من أجل المستخدم.فيما بعد،عندما يريد مستخدم مدير الحصول على إمكانية الوصول إلى الميزات المحسنة،فإن البرنامج سيحول مرةً أخرى كلمة المرور المدخلة في قيمة البعثرة(الخلط)،ويقارن تلك القيمة بكلمة المرور المخلوطة(المشفرة) في السجل. كل دالة بعثرة(خلط)للدوت نت تعتمد على كود سري.بما أن مشروع المكتبة سيعمل فقط تشفير غير مباشر،ولن يسأل أي برنامج أخر أبداً لإعادة تشفير كلمة المرور،فإننا سنستخدم فقط اسم تسجيل الدخول للمستخدم كمفتاح"سري secret". "قررت استخدام فئة البعثرة HMACSHA1،على الأغلب من أجل قدرتها على قبول مفتاح متغير الحجم variablesize. على الرغم من أنها مقررة لأن تحتوي قضايا الأمن،فإن هذا لن يكون مشكلة بالنسبة للطريقة التي سنستخدمها بها.أعني،إذا ما حاول أحدما الدخول لقاعدة البيانات فعلياً لفك تشفير كلمات المرور المخزنة في جدول UserName،فسيكون لهذه الفئة إمكانية وصول كاملة لكل شيء في نظام المكتبة سابقاً.

بالطبع،يتطلب كود التشفير مرجع إلى فضاء الأسماء System.Security.Cryptography. وسنحتاج أيضاً مرجع إلى System.Text من أجل بعض الدعم للكود.أضف عبارات Import ذات الصلة إلى أعلى ملف كود *General.vb*.

```
Imports System.Text
Imports System.Security.Cryptography
```

تحدث البعثرة(التشويش) الحقيقي لكلمة المرور في الروتين EncryptPassword، لذلك اعمل على إضافته إلى كود الوحدة البرمجية General.

```
Public Function EncryptPassword(ByVal loginID As String, ByVal passwordText As String) As String
'إدخال اسم المستخدم وكلمة المرور،تشفير كلمة المرور بحيث تصبح من صعب فك تشفيرها.
'لايوجد حد لطول كلمة المرور بما أنهاستصبح مشفرةبأي طريقة.
        Dim hashingFunction As HMACSHA1
        Dim secretKey() As Byte
        Dim hashValue() As Byte
        Dim counter As Integer
        Dim result As String = ""
'تحضير مفتاح السر.وإجباره ليكون بالحالة الكبيرة من أجل الاستقرار.
'ومن ثم حشره في مصفوفة بايت.
        secretKey = (New UnicodeEncoding).GetBytes(UCase(loginID))
'إنشاء مكونات خلط(تشفير)تستخدم SHA-1المدار
        hashingFunction = New HMACSHA1(secretKey, True)
```

```
'حساب قيمة التشفير(الخلط).سطر واحد بسيط من الكود.
        hashValue = hashingFunction.ComputeHash((New UnicodeEncoding).GetBytes(passwordText))
'قيمة البعثرة جاهزة،ولكنني أحب الأشياء في نص بسيط،
'وعندما يكون مستحيل.لنحوله إلى نص من ست عشري طويلlong hex string .
        For counter = 0 To hashValue.Length - 1
            result &= Hex(hashValue(counter))
        Next counter
'كلمات المرور المخزنة محدودة إلى 20 حرف.
        Return Left(result, 20)
    End Function
```

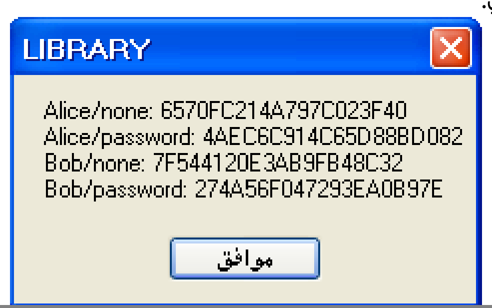
الطرق الرئيسية للتفاعل مع مزودات الأمن security providers في الدوت نت هي إما بواسطة مصفوفة بايت أو تجميع(ستريم).أثرت استخدام طريقة مصفوفة بايت،بتحويل القيم النصية القادمة من خلال الطريقة GetBytesللكائن UnicodeEncoding.حالما يتم تخزين مصفوفة بايت،قمت بتمرير معرف الدخول login IDوكلمة المرور passwordكمعاملات نسبية argumentsإلى ميزات الفئة HMACSHA1.

على الرغم من أنني أستطيع تخزين مخرجات الطريقة ComputeHashمباشرةً في حقل قاعدة البيانات،قررت تحويل النتيجة إلى رموز الآسكي ASCIIالقابلة للقراءة لذلك فإن تلك الأشياء لن تبدو مقلقة عند إصدار issuedعبارات سكول على جدول UserName.التحويل الذي قمت به أساسي basic:تحويل كل بايت byteإلى مكافئه الست عشري القابل للطباعة printable hexadecimal equivalent باستخدام دالة الفيجوال بيسك Hex .

ومن ثم سلسلة فقط النتائج مع بعضها البعض.حقل كلمة المرور لجدول UserName يحفظ فقط 20 حرف،لذلك قطعت(بترت chop off)أي شيء أطول.ومن أجل التأكد فقط أن هذه الخوارزمية تولد مخرجات معقولة، استدعيت EncryptPassword مع عدة مخرجات مختلفة.

```
MsgBox("Alice/none: " & EncryptPassword("Alice", "") & vbCrLf & "Alice/password: " & EncryptPassword("Alice",
"password") & vbCrLf & "Bob/none: " & EncryptPassword("Bob", "") & vbCrLf & "Bob/password: " &
EncryptPassword("Bob", "password"))
```

يولد الكود السابق الرسالة المبينة في الشكل التالي:



التراجع عن بعض التغيرات السابقة Undoing Some Previous Changes

تعرف الجداول UserName و GroupName و GroupActivityفي قاعدة البيانات أشكال الأمن لكل مستخدم إداري.كل مستخدم(سجل في جدول UserName)هو جزء من مجموعة أمن(سجل GroupName).كل مجموعة تتضمن إمكانية وصول إلى صفر zeroأو عدد أكبر لميزات التطبيق المحسنة،يعرف الجدول GroupActivity أي الميزات التي تتطابق مع كل سجل مجموعة أمن من جدول GroupName .

لإدارة هذه الجداول،نحتاج إلى إضافة نماذج خاصية property forms والتي تحدث حقول سجل مفرد لقاعدة البيانات.لقد كتبنا سابقاً بعض الكود الخلفي منذ فترة قليلة. عرَّف الفصل الثامن ملف *BaseCodeForm.vb*، قالب من أجل النماذج والتي تحدِث سجلات مفردة لقاعدة البيانات.ونفس الفصل قدم ملف *ListEditRecords.vb*، الفورم الرئيسي الذي يعرض قائمة بسجلات قاعدة البيانات المسبقة التعريف.محرر سجلنا من أجل كل المستخدمين usersومجموعات الأمن security groups سيستخدم ميزات في هذين النموذجين الموجودين.

عندما صممنا الكود من أجل *BaseCodeForm.vb* في الفصل الثامن،كان هدفي توضيح لك ميزات الفئة MustInheritو MustOverrideالمضمنة في الفيجوال بيسك.وهي ميزات مفيدة نوعاً ما.لسوء الحظ، إنهما فقط لايمتزجان بشكل جيد مع عناصر واجهة المستخدم،وإليك لماذا:تعمل الفيجوال أستوديو فعلياً على إنشاء حالات(نسخinstances) من نموذجك وقت التصميم design time بحيث تستطيع التفاعل معها داخل المحرر.إذا كنت تريد أن تسبر الكود المصدري،صرح عن أداة صندوق نصTextBox، فستجد كود خاص يتعامل مع تقدمة وقت التصميم للأداة.هل هو هام؟نعم،مرن؟نعم،مثالي في كل الحالات؟لا.

المشكلة ،والمشكلة هي وضعه بشكل معتدل،هل لأن فيجوال بيسك لا تستطيع إنشاء حالة فئة معرفة كـ(يجب أن تشتقMustInherit)وهذا لأنك يجب أن ترثها من خلال فئة أخرى أولاً قبل أن تنشئ حالات.ماذا يعني هذا لك؟هذا يعني إذا حاولت تصميم نموذج يشتق من قالب نموذج"يجب وراثته MustInherit"، فإن الفيجوال بيسك لن تجلب حصة واجهة المستخدم للفورم من أجل تحريرك الخاص.وما يزال بإمكانك الوصول إلى الكود المصدري للفورم،وإذا ما كانت هذه هي الطريقة التي تريد تصميم نموذج مشتقinherited form، فهذا جميل.ولكننا نبحث عن البساطة في البرمجة،لذلك سوف نستخدم بالتأكيد أدوات الفيجوال أستوديو الخاصة بالفورم لتحرير نماذجنا المرئية.

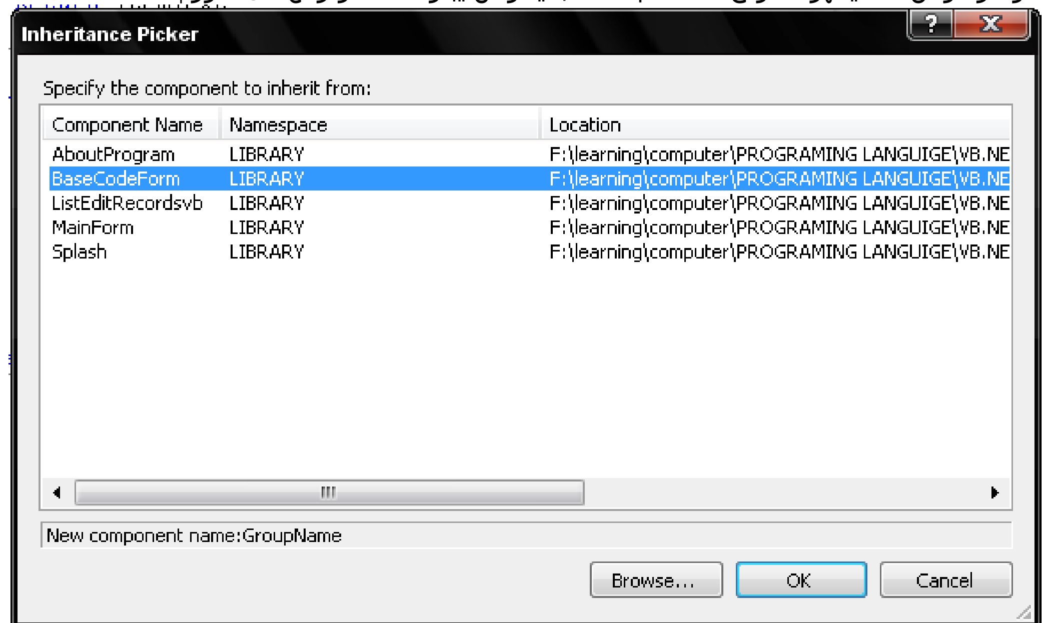
زبدة الكلام،أن علينا تغير ملف *BaseCodeForm.vb*، وإزالة الكلمات المحجوزة MustInheritوMustOverride، وعمل التعديلات المناسبة الأخرى.لقد عملت على تعديل التغيرات سابقاً.

هذا جزء من حقيقة البرمجة في الأنظمة المعقدة مثل الفيجوال أستوديو.في بعض الأحيان،حتى بعد أن تعمل جميع أبحاثك وتشبك بحذر ميزات التطبيق والتراكيب،إنك تُشغِل ضمن مصمم ما أو مترجم سلوك معين بحيث يجبرك على صنع بعض التغيير.حالما تتعلم تجنب المشاكل الرئيسية،ستجد أن هذا لايحدث كثيراً.ولكن عندما يحدث،فسيكون وقت عظيم لتغضب.

إدارة مجموعات الأمن.Managing Security Groups

لذلك،لنرجع إلى محرر سجل GroupName.لم أعمل على إضافته إلى المشروع بعد،لذلك لنعمل على إضافته الآن.لأنه سيرث من نموذج أخر في المشروع،علينا أن نسمح للفيجوال أستوديو استنساخ النموذج القاعدي بترجمة التطبيق أولاً.وهذا يتم عمله بسهولة من خلال بناء Build >> أمر قائمة بناء المكتبةBuild Library .

لإنشاء فورم form جديد،اختر مشروع Project >> أمر القائمة أضف نموذج ويندوزAdd Windows Form. عندما يظهر بند ويندوز الجديد،اختر Windows Formsمن قائمة التصنيفاتCategories، المتوبعة بواسطة Inherited Form من قائمة القوالبTemplates. أدخل في حقل الاسم **GroupName.vb**، ومن ثم انقر الزرAdd. عند يظهر نموذج ناخب الوراثةInheritance Picker form (شاهد الشكل التالي)،أختر من القائمة BaseCodeForm، وانقر موافقOK. يظهر النموذج GroupNameالجديد،ولكن يبدو على نحو رائع مثل الفورم BaseCodeForm.


Inheritance Picker

Specify the component to inherit from:

Component Name	Namespace	Location
AboutProgram	LIBRARY	F:\learning\computer\PROGRAMING LANGUIGE\VB.NE
BaseCodeForm	LIBRARY	F:\learning\computer\PROGRAMING LANGUIGE\VB.NE
ListEditRecordsvb	LIBRARY	F:\learning\computer\PROGRAMING LANGUIGE\VB.NE
MainForm	LIBRARY	F:\learning\computer\PROGRAMING LANGUIGE\VB.NE
Splash	LIBRARY	F:\learning\computer\PROGRAMING LANGUIGE\VB.NE

New component name:GroupName

Browse... OK Cancel

أضف أداتي عنوان(لصاقةLabel)،وأداتي صندوق نصTextBox، وأداتي زرButton، وأداة صندوق قائمة اختيار CheckedListBox من صندوق الأدوات، وضع خاصياتها باستخدام الإعدادات (راجع المشروع المخصص لهذا الفصل).

ولا تنسى تعديل ترتيب التنقل(مفتاح)للأدوات على الفورم،لنعمل على إضافة الكود كله دفعة واحدة،أضف الكود التالي لجسم الكود المصدري للفئة:

```
Public Class GroupName
     Private ActiveID As Integer
    Private StartingOver As Boolean
    Public Overrides Function AddRecord() As Integer
'إضافة سجل جديد.
        ActiveID = -1
        PrepareFormFields()
        Me.ShowDialog()
        If (Me.DialogResult = Windows.Forms.DialogResult.OK) Then _
            Return ActiveID Else Return -1
    End Function
     Public Overrides Function CanUserAdd() As Boolean
'اختبار المستخدم من أجل إمكانية الوصول الآن:إضافة.
        Return SecurityProfile(LibrarySecurity.ManageGroups)
    End Function
    Public Overrides Function CanUserEdit() As Boolean
'اختبار المستخدم من أجل إمكانية الوصول الآن:تحديث.
        Return SecurityProfile(LibrarySecurity.ManageGroups)
    End Function
     Public Overrides Function CanUserDelete() As Boolean
'اختبار المستخدم من أجل إمكانية الوصول الآن:حذف.
        Return SecurityProfile(LibrarySecurity.ManageGroups)
    End Function
     Public Overrides Function DeleteRecord(ByVal recordID As Integer) As Boolean
        'يريد المستخدم حذف السجل.
```

```
        Dim sqlText As String
        On Error GoTo ErrorHandler
        ' ----- Confirm with the user.
        If (MsgBox("هل تريد فعلاً حذف مجموعة الأمن؟", _
            MsgBoxStyle.YesNo Or MsgBoxStyle.Question, ProgramTitle) <> MsgBoxResult.Yes) _
            Then Return False
'التأكد من أن هذا السجل ليس قيد الاستخدام
        sqlText = "SELECT COUNT(*) FROM UserName WHERE GroupID = " & recordID
        If (CInt(ExecuteSQLReturn(sqlText)) > 0) Then
            MsgBox("لا تستطيع حذف السجل لأنه قيد الاستخدام", _
                MsgBoxStyle.OkOnly Or MsgBoxStyle.Exclamation, ProgramTitle)
            Return False
        End If
'حذف السجل.
        TransactionBegin()
        sqlText = "DELETE FROM GroupActivity WHERE GroupID = " & recordID
        ExecuteSQL(sqlText)
        sqlText = "DELETE FROM GroupName WHERE ID = " & recordID
        ExecuteSQL(sqlText)
        TransactionCommit()
        Return True
ErrorHandler:
        GeneralError("GroupName.DeleteRecord", Err.GetException())
        TransactionRollback()
        Return False
    End Function
    Public Overrides Function EditRecord(ByVal recordID As Integer) As Integer
'تحديث سجل موجود.
        ActiveID = recordID
        PrepareFormFields()
        Me.ShowDialog()
        If (Me.DialogResult = Windows.Forms.DialogResult.OK) Then _
            Return ActiveID Else Return -1
    End Function
    Public Overrides Sub FillListWithRecords(ByRef destList As ListBox, _
            ByRef exceededMatches As Boolean)
'تقديم قائمة،ملئهابجميع البنود المحددة.
        Dim sqlText As String
        Dim dbInfo As SqlClient.SqlDataReader
        On Error GoTo ErrorHandler
'تنظيف أية قيم موجودة.
        destList.Items.Clear()
        exceededMatches = False
'استخلاص القيم من قاعدة البيانات.
        sqlText = "SELECT * FROM GroupName ORDER BY FullName"
        dbInfo = CreateReader(sqlText)
        Do While dbInfo.Read
'إضافة بندواحد.
            destList.Items.Add(New ListItemData(CStr(dbInfo!FullName), CInt(dbInfo!ID)))
        Loop
        dbInfo.Close()
        dbInfo = Nothing
        Return
ErrorHandler:
        GeneralError("GroupName.FillListWithRecords", Err.GetException())
        On Error Resume Next
        If Not (dbInfo Is Nothing) Then dbInfo.Close() : dbInfo = Nothing
        Return
    End Sub
    Public Overrides Function FormatRecordName(ByVal recordID As Integer) As String
'تقديم معرف،للعودة بالاسم.
        Dim sqlText As String
        On Error GoTo ErrorHandler
'استخلاص القيمة من قاعدة البيانات.
        sqlText = "SELECT FullName FROM GroupName WHERE ID = " & recordID
        Return CStr(ExecuteSQLReturn(sqlText))
ErrorHandler:
        GeneralError("GroupName.FormatRecordName", Err.GetException())
        Return "Error"
```

```
    End Function
    Public Overrides Function GetEditDescription() As String
'إرجاع عنوان عرض نموذج القائمة
        Return "استخدم هذه الأكواد لتأسيس مجموعات أمن ضمن أي من " & _
            "المستخدمين الإداريين الذين تم وضعهم.كل مجموعة يمكن أن تستقبل " & _
            "الوصول إلى أجزاء مختلفة من النظام"
    End Function
    Public Overrides Function GetEditTitle() As String
'إرجاع القائمة من عرض العنوان.
        Return "Security Groups"
    End Function
    Private Sub PrepareFormFields()
'التعبئة بالبيانات.
        Dim sqlText As String
        Dim dbInfo As SqlClient.SqlDataReader
        Dim matchPos As Integer
        On Error GoTo ErrorHandler
'التحميل في قائمة مجموعات الأمن.
        If (ActivityList.Items.Count = 0) Then
'_ _ هذه هي المرة الأولى،التحميل من قاعدة البيانات.
            ActivityList.Items.Clear()
            sqlText = "SELECT * FROM Activity ORDER BY FullName"
            dbInfo = CreateReader(sqlText)
            Do While (dbInfo.Read)
                ActivityList.Items.Add _
                    (New ListItemData(CStr(dbInfo!FullName), CInt(dbInfo!ID)))
            Loop
            dbInfo.Close()
            dbInfo = Nothing
        Else
'تحديث الكود المتتالي.تنظيف الاختيارات.
'لاتوجد طريقة تنظيف لعمل هذا.
            Do While (ActivityList.CheckedItems.Count > 0)
                ActivityList.SetItemChecked(ActivityList.CheckedIndices(0), False)
            Loop
            If (ActivityList.Items.Count > 0) Then ActivityList.TopIndex = 0
        End If
'تنظيف البيانات الموجودة.
        StartingOver = True
        RecordFullName.Text = ""
        If (ActiveID = -1) Then Return
'تحميل القيم المخزنة في قاعدة البيانات.
        sqlText = "SELECT FullName FROM GroupName WHERE ID = " & ActiveID
        RecordFullName.Text = CStr(ExecuteSQLReturn(sqlText))
'التحميل في إعدادات الأمن الموجودة.
        sqlText = "SELECT ActivityID FROM GroupActivity WHERE GroupID = " & ActiveID
        dbInfo = CreateReader(sqlText)
        Do While (dbInfo.Read)
            matchPos = ActivityList.Items.IndexOf(CInt(dbInfo!ActivityID))
            If (matchPos <> -1) Then ActivityList.SetItemChecked(matchPos, True)
        Loop
        dbInfo.Close()
        dbInfo = Nothing
        Return
ErrorHandler:
        GeneralError("GroupName.PrepareFormFields", Err.GetException())
        On Error Resume Next
        If Not (dbInfo Is Nothing) Then dbInfo.Close() : dbInfo = Nothing
        Return
    End Sub
    Private Function SaveFormData() As Boolean
'يريد المستخدم حفظ التغيرات.الرجوع بصواب في حالة النجاح.
        Dim sqlText As String
        Dim newID As Integer = -1
        On Error GoTo ErrorHandler
'التحضير لحفظ البيانات.
        Me.Cursor = Windows.Forms.Cursors.WaitCursor
        TransactionBegin()
'حفظ البيانات.
```

```
        If (ActiveID = -1) Then
'إنشاء مدخلة جديدة.
            sqlText = "INSERT INTO GroupName (FullName) OUTPUT INSERTED.ID VALUES (" & _
                DBText(Trim(RecordFullName.Text)) & ")"
            newID = CInt(ExecuteSQLReturn(sqlText))
        Else
'تحديث المدخلة الموجودة.
            newID = ActiveID
            sqlText = "UPDATE GroupName SET FullName = " & _
                DBText(Trim(RecordFullName.Text)) & " WHERE ID = " & ActiveID
            ExecuteSQL(sqlText)
        End If
'تنظيف أي إعدادات أمن موجودة.
        sqlText = "DELETE FROM GroupActivity WHERE GroupID = " & newID
        ExecuteSQL(sqlText)
        ' ----- Save the selected security settings.
'حفظ إعدادات الأمن المختارة.
        For Each itemChecked As ListItemData In ActivityList.CheckedItems
            sqlText = "INSERT INTO GroupActivity (GroupID, ActivityID) VALUES (" & _
                newID & ", " & itemChecked.ItemData & ")"
            ExecuteSQL(sqlText)
        Next itemChecked
'إتمام جميع التغيرات.
        TransactionCommit()
        ActiveID = newID
'يمكن لهذا التغير أن يؤثر على هذا المستخدم.
        If (LoggedInGroupID = ActiveID) Then ReprocessSecuritySet()
'_ _ _ بنجاح.
        ActiveID = newID
        Me.Cursor = Windows.Forms.Cursors.Default
        Return True
ErrorHandler:
        Me.Cursor = Windows.Forms.Cursors.Default
        GeneralError("GroupName.SaveFormData", Err.GetException())
        TransactionRollback()
        Return False
    End Function
    Private Function ValidateFormData() As Boolean
'_ _ تأكد من أن المستخدم يزود بالبيانات الصحيحة.الرجوع بصواب في حالة النجاح.
        Dim sqlText As String
        On Error GoTo ErrorHandler
        ' ----- Name is a required field.
'_ _ الاسم هو الحقل المطلوب.
        If (Trim(RecordFullName.Text) = "") Then
            MsgBox("Please supply the security group name.", _
                MsgBoxStyle.OkOnly Or MsgBoxStyle.Exclamation, ProgramTitle)
            RecordFullName.Focus()
            Return False
        End If
'_ _ تأكد من الاسم ليس قيد الاستخدام سابقاً.
        sqlText = "SELECT COUNT(*) FROM GroupName WHERE UPPER(FullName) = " & _
            DBText(UCase(Trim(RecordFullName.Text)))
        If (ActiveID <> -1) Then sqlText &= " AND ID <> " & ActiveID
        If (CInt(ExecuteSQLReturn(sqlText)) > 0) Then
            MsgBox("A record already exists with the name '" & Trim(RecordFullName.Text) & _
                "'.", MsgBoxStyle.OkOnly Or MsgBoxStyle.Exclamation, ProgramTitle)
            RecordFullName.Focus()
            Return False
        End If
'_ _ بنجاح.
        Return True
ErrorHandler:
        GeneralError("GroupName.ValidateFormData", Err.GetException())
        Return False
    End Function
    Private Sub GroupName_Activated(ByVal sender As Object, ByVal e As System.EventArgs) Handles
Me.Activated
'_ _ العودة بالتركيز إلى الحقل الرئيسي.
        If (StartingOver) Then RecordFullName.Focus()
```

```
        StartingOver = False
    End Sub
     Private Sub RecordFullName_Enter(ByVal sender As Object, ByVal e As System.EventArgs) Handles
RecordFullName.Enter
' _ _ علم كامل النص.
        RecordFullName.SelectAll()
    End Sub
    Private Sub ActOK_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
ActOK.Click
' _ _ احفظ التغيرات وارجع إلى نموذج الملخص.
        If (ValidateFormData() = False) Then Return
        If (SaveFormData() = False) Then Return
        Me.DialogResult = Windows.Forms.DialogResult.OK
    End Sub
End Class
```

تتضمن الفئة عضوين خاصين.يحفظ ActiveIDرقم ID لسجل GroupName لقاعدة البيانات المعروض،أو-1 عند تحرير سجل جديد.العلامة StartingOverأكثر أهمية نوعاً ما.تذكر أننا نستخدم نموذج ملخص متشارك لعرض جميع سجلات GroupNameالمدخلة سابقاً.للسماح لهذه الفورم الشاملةListEditRecords.vb ، من العمل مع محررات السجلات المختلفة،نمرر حالة من نموذج التفاصيل(GroupName.vb في هذه الحالة)إلى نموذج الملخص:

```
ListEditRecordsvb.ManageRecords(New LIBRARY.GroupName)
```

ضمن كود النموذج ListEditRecordsvb، يتم استخدام حالة من GroupNameمرةً بعد مرةً،كل مرة يريد المستخدم إضافة أو تحديث سجل قاعدة البياناتGroupName. إذا حدث المستخدم سجل واحد،ومن ثم حاول تحديث واحد أخر،فإن المتبقي من السجل الأول سيبقى موجود في حقول نموذج التفاصيل.وهكذا،سيكون علينا تنظيفها كل مرة نضيف أو نحدث سجل مختلف.يساعد الدليل StartingOverبهذه العملية وذلك بإعادة وضع التركيز إلى حقل نموذج التفاصيل الأول في حدث Activatedللنموذج .

```
Private Sub GroupName_Activated(ByVal sender As Object, ByVal e As System.EventArgs) Handles
Me.Activated
' _ _ العودة بالتركيز إلى الحقل الرئيسي.
        If (StartingOver) Then RecordFullName.Focus()
        StartingOver = False
    End Sub
```

الطريقة الخاصة PrepareFormFieldsتعمل التنظيف الحقيقي وتخزن البيانات مع كل استدعاء لتحديث أو إضافة جديدة.من أجل السجلات الجديدة، من البساطة تنظيف كل البيانات المدخلة على الفورم.عند تحديث سجل موجود،فإنه يستخلص البيانات ذات الصلة من قاعدة البيانات،ويخزن القيم المحفوظة في حقول متنوع على الفورم.العبارات التالية تعرض اسم المجموعة المخزنة في حقل RecordFullName،أو أداة صندوق نصTextBox .

```
' تحميل القيم المخزنة في قاعدة البيانات.
         sqlText = "SELECT FullName FROM GroupName WHERE ID = " & ActiveID
         RecordFullName.Text = CStr(ExecuteSQLReturn(sqlText))
```

معظم الروتينات في نموذج GroupName توفر إعادة قيادة overridesبسيطة من أجل الأعضاء القاعدية لفئة BaseCodeForm.الطريقة CanUserAdd والتي تعود ببساطة بـخطأ False في الفئة القاعدية،تتضمن منطق حقيقي في الفئة المشتقة.إنها تستخدم المصفوفة SecurityProfileالتي أضفناها سابقاً لتحديد أي مستخدم حالي مسموح له إضافة سجلات مجموعةgroup records .

```
Public Overrides Function CanUserAdd() As Boolean
'          اختبار المستخدم من أجل إمكانية الوصول الآمن:إضافة.
         Return SecurityProfile(LibrarySecurity.ManageGroups)
    End Function
```

إذا نظرت إلى الكود المضاف،ستجد إعادة قيادة overridesلجميع أعضاء BaseCodeForm ماعدا الطرق UsesSearchو SearchForRecord .الفئة المشتقة تقبل الفعل الافتراضي لهذين العضوين.

يضيف المستخدمadds، يحدثedits، ويحذف deletesسجلات اسم مجموعة من خلال إعادة قيادة AddRecord،وEditRecord و DeleteRecord، على الترتيب،لكل استدعاء من قبل الكود في فورم ListEditRecords. إليك كود EditRecord.

```
Public Overrides Function EditRecord(ByVal recordID As Integer) As Integer
' تحديث سجل موجود.
        ActiveID = recordID
        PrepareFormFields()
        Me.ShowDialog()
        If (Me.DialogResult = Windows.Forms.DialogResult.OK) Then _
            Return ActiveID Else Return -1
    End Function
```

بعد تخزين معرف ID السجل للتحديث في الحقل الخاصActiveID، يحمل الكود البيانات من خلال الطريقةPrepareFormFields، ويحث المستخدم على تحديث السجل باستدعاءMe.ShowDialog. ينتظر الفورم حتى يضع كود ما أو أداة خاصية الفورمDialogResult. وهذا يتم عمله في الحدث

ActOK_Click،وأيضاً من خلال الخاصية DialogResultللزر ActCancel، والتي ستعمل الفيجوال أستوديو على إسناده إلى الفورم بشكل آلي عندما ينقر المستخدم الزر ActCancel .

إن الروتين AddRecordيشبه كثيراً الروتين EditRecord، ولكنه يسند -1 إلى العضو ActiveIDليشير لسجل جديد.الروتين DeleteRecordأكثر تعقيداً،ويستخدم بعض كود قاعدة البيانات الذي كتبناه في الفصل السابق(راجع كود الدالة DeleteRecord)

بعد تأكيد الحذف من قبل المستخدم،يحدد تفحص سريع فيما إذا المجموعة group تزال قيد الاستخدام في مكان في جدول UserName. إذا كان كل شيء تم مراجعته على مايرام،يتم حذف السجل باستخدام عبارة سكول DELETE.بما أننا نحتاج إلى حذف البيانات في جدولين،فقد ضمنته كله في مداولة transaction.إذا ما حدث خطأ،فإن معالج الخطأ عند نهاية الروتين سيتراجع عن المداولة من خلال TransactionRollback.

ملاحظة:تحذير تكامل قاعدة البيانات.Database Integrity Warning

إذا كان لديك خلفية في تطوير قاعدة البيانات،فقد تكون قد رأيت سابقاً الخلل في كود الحذفdelete. على الرغم من أنني أخذت وقتي في إثبات أن السجل ليس قيد الاستخدام قبل حذفه،من المحتمل أن يستخدمه بعض المستخدمين الآخرين بين الوقت الذي أختبر فيه استخدام السجل والوقت الذي أحذفه فعلياً. بالاعتماد على إعداد(تركيب) الكود وقاعدة البيانات الذي جلبته حتى الآن،ستكون هناك مشكلة حقيقية.عندما صممت هذا النظام،توقعت أن أمين مكتبي مفرد سيدير مهمات إدارية كهذه،لذلك لم أقلق حول تعارضات conflictsكهذه .

إذا كنت مهتم حول احتمالات حذف سجلات قيد الاستخدام من خلال كود مثل هذا،تستطيع تمكين التكامل المرجعي *referential integrity* على العلاقات في قاعدة البيانات.لقد أسست علاقة بين الحقول GroupName.ID وUserName.GroupID، ولكن كانت من أجل الأهداف التعليمية فقط.تستطيع إعادة تركيب هذه العلاقة ليملك سكول سرفر القدرة على إجبار العلاقة بين الجداول.إذا ما عملت هذا،فلن يكون من الممكن حذف سجل قيد الاستخدام، وسيحدث خطأ في البرنامج عندما تحاول فعل هذا.إن هذا يبدو جيد،وهو كذلك،ولكن الإفراط في استخدام التكامل المرجعي يمكن أن يبطئ إمكانية الوصول لبياناتك.وسأترك هذا التركيب(الإعداد)اختياري بالنسبة لك.

عندما يتم عمل تغيرات من قبل المستخدم إلى السجل،فإن النقر على الزر"موافق OK "يدفع البيانات المخرجة من قاعدة البيانات إليها مرةً أخرى. يتحقق معالج الحدث ActOK_Click من البيانات،ويحفظها.

```
'_ _ احفظ التغيرات وارجع إلى نموذج الملخص.
        If (ValidateFormData() = False) Then Return
        If (SaveFormData() = False) Then Return
        Me.DialogResult = Windows.Forms.DialogResult.OK
```

تعمل الطريقة ValidateFormDataبعض الاختبارات البسيطة للتحقق من البيانات،مثل الطلب من المستخدم إدخال اسم مجموعة الأمن security group name،وهذا مميز.إذا ما بدا كل شيء على مايرام،يبني الروتين SaveFormDataعبارات سكول التي تحفظ البيانات(راجع كود الدالة SaveFormData).تأكد من مراجعة الروتينات الأخرى في نموذجGroupName، فهي موجودة لدعم وتحسين خبرة المستخدم.

إدارة المستخدمين.Managing Users

نحتاج أيضاً إلى فورم لإدارة سجلات جدولUserName. بما أن كود تلك الفورم بشكل عام يتبع ما رأيناه سابقاً في نموذجGroupName، فلن أثقل عليك بالتفاصيل.لقد عملت على إضافة *UserName.vb* لمشروعك،ولكن لمنع الأخطاء في كودك بينما تكون في وسط التطوير،فقد عملت على تعطيله. ولتمكينه،اختر هذا الملف في نافذة مستكشف الحلولSolution Explorer. ومن ثم في نافذة الخصائصProperties، غير خاصية Build Action من Noneإلى Compile.

الكود الوحيد المهم في هذه الفورم والذي هو مختلف نوعاً ما عن فورم GroupName هو معالجة كلمة المرورpassword. وللحفاظ على الأشياء سرية قدر الإمكان،فلن أحمل فعلياً كلمة المرور المحفوظة ضمن حقل كلمة المرور على الفورم. ولن أعمل أية طريقة جيدة منذ أن عملت على تخزين نسخة(إصدار) الخلط hashed version في قاعدة البيانات.بما أنني استخدم معرف تسجيل الدخول Login ID للمستخدم كمفتاح سري عند تشفير كلمة المرور،يجب عليّ أن أعيد توليد كلمة المرور passwordولو غيّر المستخدم معرف تسجيل الدخولLogin ID. يحفظ الحقل الخاص OrigLoginID نسخة من معرف تسجيل الدخول Login ID عند فتح الفورم للمرة الأولى، ويتفحص من أجل أي تغيرات عند إعادة حفظ السجل.إذا ما حدثت تغيرات،فإنه يعيد توليد كلمة المرور.

```
passwordResult = EncryptPassword(Trim(RecordLoginID.Text), Trim(RecordPassword.Text))
```

استخدام نماذج التحرير UserName وGroupNameيتطلب بعض الكود الإضافي في الفورم الرئيسيmain form. أضفه إلى جسم الكود المصدري للفورم الرئيسي:

```
Private Sub AdminLinkGroups_LinkClicked(ByVal sender As Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles AdminLinkGroups.LinkClicked
        ' ----- Let the user edit the list of security groups.
'        دع المستخدم يحرر قائمة مجموعات الأمن.
        If (SecurityProfile(LibrarySecurity.ManageGroups) = False) Then
            MsgBox(NotAuthorizedMessage, MsgBoxStyle.OkOnly Or _
                MsgBoxStyle.Exclamation, ProgramTitle)
            Return
        End If
        ' ----- Edit the records.
'        تحرير السجلات.
        ListEditRecords.ManageRecords(New Library.GroupName)
        ListEditRecords = Nothing
```

```
    End Sub
    Private Sub AdminLinkUsers_LinkClicked(ByVal sender As Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles AdminLinkUsers.LinkClicked
        ' ----- Let the user edit the list of administrative users.
دع المستخدم يحرر قائمة المستخدمين الإداريين.      '
        If (SecurityProfile(LibrarySecurity.ManageUsers) = False) Then
            MsgBox(NotAuthorizedMessage, MsgBoxStyle.OkOnly Or _
                MsgBoxStyle.Exclamation, ProgramTitle)
            Return
        End If
        ' ----- Edit the records.
تحرير السجلات.      '
        ListEditRecords.ManageRecords(New Library.UserName)
        ListEditRecords = Nothing
    End Sub
```

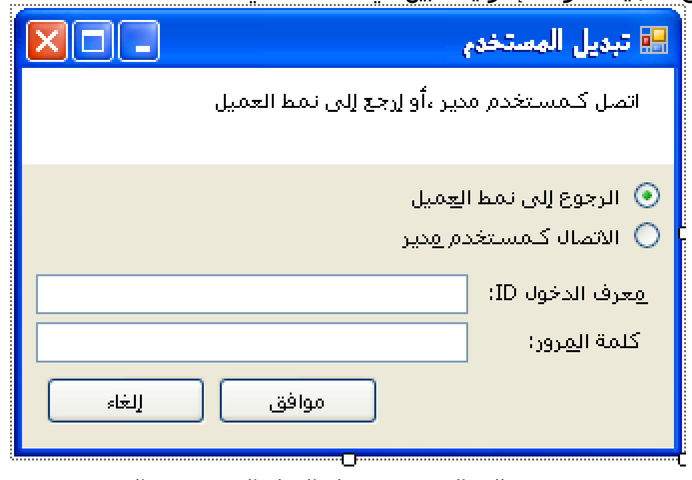
الأدوات AdminLinkGroups وAdminLinkUsersهي لصاقات وصل نمطية الويب web-style link labels والتي أضفناها للبرنامج من عدة فصول سابقة.الحدث LinkClickedوليس الحدث Clicked هو من يطلق العرض لمحرر الكود.إليك الكود الذي يحرر جدول GroupName.

```
Private Sub AdminLinkGroups_LinkClicked(ByVal sender As Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles AdminLinkGroups.LinkClicked
        ' ----- Let the user edit the list of security groups.
'دع المستخدم يحرر قائمة مجموعات الأمن.
        If (SecurityProfile(LibrarySecurity.ManageGroups) = False) Then
            MsgBox(NotAuthorizedMessage, MsgBoxStyle.OkOnly Or _
                MsgBoxStyle.Exclamation, ProgramTitle)
            Return
        End If
        ' ----- Edit the records.
' تحرير السجلات.
        ListEditRecords.ManageRecords(New Library.GroupName)
        ListEditRecords = Nothing
    End Sub
```

تجربة كل مستخدم.Per-User Experience

والآن بما أن لدينا كل كود دعم الأمن المضاف إلى المشروع،نستطيع البدء باستخدام هذه الميزات لتغير تجربة التطبيق من أجل الزبائن والمدراء. ليس من الخُلق إغراء الناس بالقوة الهائلة،لذلك من الأفضل إخفاء هذه الميزات والتي هي غير قابلة للوصول بالنسبة للعملاء المستخدمين المتواضعين وبشكل أساسي أقل قوة.

أولاً،لنوفر القوة المتنوعة بإضافة نموذج تسجيل دخول الإداري،المبين في الشكل التالي.



لقد عملت سابقاً على إضافة النموذج ChangeUser.vb إلى المشروع.معظم العمل الصعب في الفورم يحدث في معالج الحدثActOK_Click. إذا ما اختار المستخدم"الرجوع إلى نمط العميل"،جميع قيم الأمن يتم تنظيفها،والفورم الرئيسي تخفي معظم الميزات(من خلال كود سيضاف فيما بعد)

```
LoggedInUserID = -1
            LoggedInUserName = ""
            LoggedInGroupID = -1
            ReprocessSecuritySet()
```

تعمل هذه الفورم على الاتصال إلى التطبيق من خلال الفورم الرئيسي وذلك من خلال الحدثActLogin_Click. افتح ملف الفورم الرئيسي MainForm.vb، انقر مزدوج على زر Loginفي الزاوية العلوية اليسارية،وأضف الكود التالي لحدث النقر الذي يظهر.

```
Private Sub ActLogin_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
ActLogin.Click
```

```
' حث المستخدم على اختيار نمط العميل أو النمط الإداري
        ShowLoginForm()
    End Sub
```

أضف الكود التالي للفورم أيضاً:

```
Private Sub ShowLoginForm()
        ' حث المستخدم على اختيار نمط العميل أو النمط الإداري
        Dim userChoice As Windows.Forms.DialogResult
        userChoice = ChangeUser.ShowDialog()
        ChangeUser = Nothing
        If (userChoice = Windows.Forms.DialogResult.OK) Then UpdateDisplayForUser()
    End Sub
```

لنمكن أيضاً المفتاح F12 ليتصرف كمطلق للدخولlogin .أضف الكود التالي إلى عبارة "اختر حالة Select Case " في معالج حدث MainForm_KeyDown.

```
Case Keys.F12
' حث المستخدم من أجل نمط العميل أو النمط الإداري.
  ShowLoginForm()
  e.Handled = True
```

يستدعي الروتين ShowLoginFormطريقة أخرى، UpdateDisplayForUser،والتي تخفي وتظهر عناصر العرض المتنوعة على الفورم الرئيسي بالاعتماد على شكل الأمن security profile للمستخدم الحالي.أضفها إلى كود فئة النموذج الرئيسيMainForm. وهو ينظر بشكل أساسي إلى المتغير LoggedInUserID،وإذا ما تم وضعه إلى -1،فإنه يخفي جميع الأدوات بالنسبة للميزات المتقدمة.

```
Private Sub UpdateDisplayForUser()
' تحديث العرض بالاعتماد على إعدادات المستخدم
        Dim sqlText As String
        If (LoggedInUserID = -1) Then
' إعداد العرض من أجل نمط العميل
            ActiveMode.Text = "مرحباً بك زبون المكتبة"
            LabelTasks.Visible = False
            LineTasks.Visible = False
            PicCheckOut.Visible = False
            ActCheckOut.Visible = False
            PicCheckIn.Visible = False
            ActCheckIn.Visible = False
            PicAdmin.Visible = False
            ActAdmin.Visible = False
            PicProcess.Visible = False
            ActProcess.Visible = False
            PicReports.Visible = False
            ActReports.Visible = False
            ActLogin.Visible = True
' بنود البحث بشكل افتراضي
            TaskLibraryItem()
        Else
' إعداد العرض من أجل النمط الإداري
            Try
                sqlText = "SELECT FullName FROM UserName WHERE ID = " & LoggedInUserID
                ActiveMode.Text = "Administrative User: " & CStr(ExecuteSQLReturn(sqlText))
            Catch ex As Exception
                GeneralError("MainForm.UpdateDisplayForUser", Err.GetException())
                ActiveMode.Text = "Administrative User: Unknown"
            End Try
            LabelTasks.Visible = True
            LineTasks.Visible = True
            PicCheckOut.Visible = True
            ActCheckOut.Visible = True
            PicCheckIn.Visible = True
            ActCheckIn.Visible = True
            PicAdmin.Visible = True
            ActAdmin.Visible = True
            PicProcess.Visible = True
            ActProcess.Visible = True
            PicReports.Visible = True
            ActReports.Visible = True
            ActLogin.Visible = True
        End If
    End Sub
```

حالياً،عندما نشغل التطبيق،جميع الميزات المتقدمة تظهر،حتى ولو لم يكن قد زود مدير بالمعرف ID أو كلمةالمرورpassword. استدعاء UpdateDisplayForUser عندما يظهر الفورم الرئيسي للمرة الأولى يحل المشكلة.أضف الكود التالي إلى نهاية الطريقة MainForm_Load.

```
' التحضير للمستخدم العميل
        UpdateDisplayForUser()
```

التحديث الأخير(عملياً،خمس تحديثات)يتضمن وضع حدود المقاطع الرئيسية للفورم لضبط المدراء المرخص لهم.على سبيل المثال،فقط المدراء المرخص لهم بتشغيل التقارير يجب أن يكونوا قادرين على الوصول إلى لوحة التقارير على الفورم الرئيسي.أوجد الطريقة TaskReportsفي الفورم الرئيسي،وأوجد السطر الذي يعرض اللوحة.

```
PanelReports.Visible = True
```

بدل هذا السطر بالكود التالي.

```
If (SecurityProfile(LibrarySecurity.RunReports)) Then PanelReports.Visible = True
```

نحتاج لعمل نفس الشيء في الطرق TaskCheckOut، ، TaskAdmin وTaskProcess. في كل حالة بدل السطر الذي يقرأ:

```
Panel???.Visible = True
```

بالكود الذي يختبر إعدادات الأمن قبل إظهار اللوحة.

فبدل السطر التالي `PanelCheckOut.Visible = True` بالكود التالي من أجل TaskCheckOut.

```
If (SecurityProfile(LibrarySecurity.CheckOutItems)) Then PanelCheckOut.Visible = True
```

وبدل السطر التالي `PanelCheckIn.Visible = True` بالكود التالي من أجل TaskCheckIn.

```
If (SecurityProfile(LibrarySecurity.CheckInItems)) Then PanelCheckIn.Visible = True
```

وبدل السطر التالي `PanelAdmin.Visible = True` بالكود التالي من أجل الطريقة TaskAdmin.

```
If (SecurityProfile(LibrarySecurity.AdminFeatures)) Then PanelAdmin.Visible = True
```

بدل السطر التالي `PanelProcess.Visible = True` بالكود التالي من أجل الطريقةTaskProcess .

```
If (SecurityProfile(LibrarySecurity.DailyProcessing)) Then PanelProcess.Visible = True
```

شَغِّل البرنامج وسترى أن بدايته تبدو كتطبيق حقيقي.إذا حاولت الوصول إلى الميزات المحسنة،جرب معرف تسجيل الدخول " admin "وبدون كلمة مرور. تستطيع تغير ذلك من خلال نموذج UserNameإذا أردت.

بما أنه لدينا طريقة لتأمين الوصول إلى البيانات والميزات بالنسبة لمشروع المكتبة،لذلك لننتقل إلى الفصل التالي ونبدأ بالتركيز على البيانات،النقطة المركزية لأي تطبيق عمل.

إعادة التعريف والامتدادات **Overloads and Extensions**

بدايةً، إن الفيجوال بيسك تدعم إعادة تعريف*operator overloading* المعاملات مثل الجمع(+).هذا الفصل سيريك كيف تستطيع توجيه عملية تحسين البناء الجسدي من أجل معاملات الفيجوال بيسك المتنوعة.وسأقدم لك أيضاً طرق الامتداد(التوسيع *extension*)،والتي تتيح لك بشكل مشابه تحسين الفئات،حتى ولو لم يكن عليك الوصول إلى الكود المصدري الأصلي لهذه الفئات.

ماذا نعني بإعادة تعريف المعامل؟**What Is Operator Overloading?**

تتيح عملية إعادة تعريف معامل لكودك تحسين معاملات الفيجوال بيسك الأساسية وتمنحهم إمكانيات غير متوفرة لديهم سابقاً بواسطة المترجم.إعادة التعريف لا تغير البناء المستعمل عند توظيف المعاملات،ولكنها تغير نوع الكائنات التي يديرها كل معامل.على سبيل المثال،معامل الضرب(*)عادة يتفاعل مع الأعداد فقط،ولكن تستطيع تضخيمه للعمل مع فئة مثل "النحلة الطنانة Bumblebee " التي تخصصها أنت.

```
Dim swarm As Bumblebee
   Dim oneBumblebee As New Bumblebee
   Dim twoBumblebee As New Bumblebee
swarm = oneBumblebee * twoBumblebee
```

المعنى الذي تطبقه على المعامل المعاد تعريفه عائد لك.على الرغم من أنك عندما تريد أن تحتفظ بالنوع المضاف لمعامل الجمع (الإضافة) عند إعادة تعريفه،فليس عليك عمل هذا.في الحقيقة،تستطيع إعادة تعريف معامل الإضافة بحيث يعمل على طرح قيمة من أخرى.ولكنني سأطردك من العمل عندي إذا عملت ذلك،هذا فقط كي تعلم أنه ممكن.

جميع ميزات إعادة تعريف معامل مرتبطة بشكل مباشر بواحد أو أكثر من فئاتك.تبدو الميزات المعاد تعريفها مثل أعضاء دالة قياسية تقريباً،وتظهر كأعضاء لفئاتك.

تتضمن الفيجوال بيسك نوعين من المعاملات:أحادي *unary*والثنائية*binary*، معرفة تبعاً لعدد العوامل(أطراف المعامل) المميزة بواسطة المعامل.تقبل المعاملات الأحادية طرف(عامل) واحد فقط،والذي يظهر دائماً على يمين اسم المعامل أو الرمزsymbol .فمنطق المعامل "ليس Not "هو معامل أحادي:

```
oppositeValue = Not originalValue
```

تقبل المعاملات الثنائية عاملين(طرفين)واحد على كل جهة من المعامل.فمعامل الضرب هو معامل ثنائي:

```
ten = two * five
```

طبيعة معامل ما هي ما يعمله مباشرةً، في الحقيقة يتم استبدال المعامل وعوامله(عامله)المدخلة بالنتيجة المحسوبة.فمثلاً التعبير(10/5)يتم استبداله بالنتيجة المحسوبة(2)،وهذه النتيجة يتم استخدامها لإتمام عبارة ما أو تعبير عملية أصلية يظهر ضمنه المعامل وعوامله.فهو يعمل تماماً كدالة:

```
'السطرين التاليين يعملان نفس الشيء
' النتيجة المحسوبة في المتغيرtheAnswer
theAnswer = 2 * 5
theAnswer = DoubleIt(5)
```

ليكون معامل جاهز لإعادة التعريف،غير تفكيرك لترى المعاملات كدوال.انظر عبر حدود بيئة معاملك،وافتح عقلك إلى حقيقة أن المعاملات والدوال هي واحدة.

تعريف إعادة تعريف المعاملات في الفيجوال بيسك كما يلي،إذا كان عليك ترجمة تعريف الضرب إلى دالة فيجوال بيسك،يمكن أن تبدو كالتالي:

```
Public Shared Function *(ByVal firstOperand As Integer, ByVal secondOperand As Integer) As Integer
```

المعامل(*)يصبح اسم الدالة،مع العوامل operandsالتي تلعب دور وسيطات الدالة،أخيراً ،توليد القيمة المعروضة من خلال القيمة المعادة بالدالة.على الرغم من أن المعاملات لا يتم تعريفها كدوال بهذه الطريقة في الفيجوال بيسك،ولكن إعادة تعريف هذه المعاملات يتم تعريفها كدوال.

لإعادة تعريف معامل الضرب في فئة Bumblebeeالخيالية،نستخدم الكلمة المحجوزة Operatorلتعريف"دالة الضرب multiplication function" من أجل عوامل الفئة Bumblebee.

```
Partial Class Bumblebee
   Public Shared Operator *(ByVal operand1 As Bumblebee, _
   ByVal operand2 As Bumblebee) As Bumblebee
      ' ----- Multiply two bumblebees together.
      Dim finalResult As New Bumblebee
      ' ----- Add special "multiplication" code here, then...
      Return finalResult
   End Operator
End Class
```

والآن عندما تضرب حالتي Bumblebeeمع بعضهما بمعامل الضرب،تميز الفيجوال بيسك النموذج"عامل1*عامل2" كمطابقة لإعادة تعريف معامل الضرب مع معاملين نسبيين من نوع Bumblebee،وتستدعي هذه الفئة بالاعتماد على دالة المعامل للحصول على النتيجة.

جميع التصريحات عن المعاملات يجب أن تتضمن الكلمة المحجوزة Public والكلمة المحجوزةShared .فإذا لم تكن متشاركة،shared سيكون على الفيجوال بيسك طلب إنشاء حالة إضافية من الفئة من أجل الوصول لكود إعادة تعريف المعامل فقط،وذلك لن يكون فعال جداً.

ما الذي تستطيع إعادة تعريفه؟**What Can You Overload?**

تستطيع إلى حد ما إعادة تعريف أي معاملات قياسية للفيجوال بيسك(ما عدا المعامل Is والمعاملIsNot)زائد عدة ميزات أخرى.يشرح هذا الفصل كل معامل قابل لإعادة التعريف،مصنف بواسطة النوع العام.كل مقطع يتضمن جدول بالمعاملات.لإعادة تعريف معامل في فئة،استخدم الاسم في عمود المعامل كاسم للدالة.إذا كان هناك معامل مسمى XX،ستكون عبارة المعامل المطابقة كما يلي:

```
Public Shared Operator XX(...)
```

المعاملات الرياضية **Mathematical Operators**

تعرف الفيجوال بيسك 10معاملات رياضية أو معاملات رياضية مستعارة(شبه رياضية) pseudo-mathematical .جميعها لمعاملة الأعداد ما عدا واحد،المعامل الباقي هو مسلسل النصوصstring concatenation (&) ولكن يبدو مثل باقي المعاملات الرياضية في بناءه واستخدامه.
معاملين من المعاملات،الزائد plus (+) والناقص minus (-)،هما معاملات أحادية unaryوثنائيةbinary .إشارة الناقص minus sign تعمل كمعامل"نفي negation" أحادي كما في("5-")وكمعامل "طرح subtraction" ثنائي(البناء "2-5"المعروف).عندما إعادة تعريف هذه المعاملات،الاختلاف الموضوع في عدد المعاملات النسبية المضمنة في توقيع المعامل يحدد أي منها أحادي أم ثنائي.

```
Public Shared Operator -(ByVal operand1 As SomeClass) As SomeClass
    ' ----- This is the unary "negation" version.
End Operator
```

تجدول القائمة التالية المعاملات الرياضية التي تدعم إعادةالتعريف:

المعاملOperator	النوعType	التعليقComments
+	Unary	المعامل"زائد" الأحادي.تستطيع استخدام هذا المعامل مع الأعداد،كما في(5+).ولكن إذا أدخلت هذه القيمة في الفيجوال أستوديو،فإنه سيتم تجريد (إزالة) معامل الزائد،بما أنه يعتبر فائض(زائد عن الحاجة).مهما يكن،إذا عملت على إعادة قيادة هذا المعامل على فئة خاصة بك،فإن الفيجوال أستوديو سيحفظ الصيغة الأحادية لهذا المعامل عند الاستخدام في الكود.

```
' ----- افرض أن المعامل+ قد تم إعادة تعريفه...
Dim oneBuzz As New Bumblebee
Dim moreBuzz As Bumblebee = +oneBuzz
```

المعاملOperator	النوعType	التعليقComments
		بما أن هذا المعامل أحادي،يتضمن فقط معامل نسبي واحد عند تعريف طريقة المعاملOperator .
+	Binary	معامل الإضافة(الجمع)القياسي.تذكر،فقط لأن المعامل يدعى معامل"إضافة"لايعني أن عليك حفظ ذلك المفهوم.مهما يكن،عليك المحاولة في إعادة تعريف المعاملات كأقرب ما يمكن لمعناه الأصلي قدر الإمكان،الفيجوال بيسك نفسها تعيد قيادة هذا المعامل لتسمح له بالتصرف قليلاً مثل معامل سلسلة النصوص.
-	Unary	هذا معامل"سلبي(نفي)negation)"أحادي والذي يأتي تماماً قبل قيمة أو تعبير.
-	Binary	معامل الطرحsubtraction .
*	Binary	معامل الضرب multiplication.
/	Binary	معامل القسمة القياسيstandard division operator .
\	Binary	معامل القسمة الصحيحةinteger division operator. تذكر أنك لست بحاجة للاحتفاظ بأي شيء "عددي صحيحinteger " في هذا المعامل إذا لم يوافق احتياجات فئتك.
^	Binary	معامل الدليل أو الأس exponentiation ("القوة من to the power of ")
Mod	Binary	معامل الباقيmodulo، بعض الأحيان يدعى معامل الباقي remainder operator لأنه يعود بالباقي من عملية القسمة.
&	Binary	معامل سلسلة النصوص string concatenation operator.

معاملات المقارنة Comparison Operators

تتضمن الفيجوال سبع معاملات مقارنة أساسية،معظمها يتم استخدامه في عبارة Ifو التعابير المشابهة والتي تتطلب حساب شروط منطقية *Boolean* . طرق المعامل Operatorمن أجل معاملات المقارنة هذه لها نفس البناء الذي تم استخدامه في المعاملات الرياضية،ولكن معظمها يجب أن يتم معالجته بشكل زوجي. على سبيل المثال،إذا أعدت تعريف المعامل(أقل من>)،تطلب منك الفيجوال بيسك إعادة تعريف المعامل(أكبر من<)أيضاً ضمن نفس الفئة،ومن أجل نفس توقيع المعامل النسبي argument signature.
جميع معاملات المقارنة هي معاملات منطقية.على الرغم من أنك تستطيع تبديل نوع البيانات للمعاملات النسبية الممررة للمعامل،ويجب أن تعود كلها بقيمة منطقية.

```
Public Shared Operator <=(ByVal operand1 As SomeClass, ByVal operand2 As SomeClass) As Boolean
    ' ----- The <= operator returns a Boolean result.
End Operator
```

القائمة التالية تجدول سبع معاملات مقارنة أساسية والتي تستطيع إعادة تعريفها.وكل مدخلة تتضمن قيمة"المصاحبbuddy " والتي تحدد المعامل الموافق والذي يجب أن يتم إعادة تعريفه.

المعاملOperator	المصاحبBuddy	التعليقComments
=	<>	يقارن معامل المساواة عاملين operandsمن أجل التساويequivalence، ويعود بصواب Trueإذا كانا متساويان.
<>	=	يقارن معامل عدم المساواة عاملين من أجل عدم التساوي،ويعود بصواب إذا كانا غير متطابقين.
<	>	يعود المعامل أقل من بصواب إذا كان العامل الأول "أقل منless than "الثاني.
>	<	يعود المعامل أكبر من بصواب إذا كان العامل الأول"أكثر منgreater than "الثاني.
<=	>=	يعود المعامل أقل من أو يساوي إلى،بصواب إذا كان العامل الأول "أقل من أو يساوي لless than or equal to "الثاني.
>=	<=	يعود المعامل أكثر من أو يساوي ل،بصواب إذا كان العامل الأول "أكبر من أو يساوي لgreater than or equal to "للثاني.

معامل المقارنة السابع هوLike .في الفيجوال بيسك القياسي،يعمل هذا المعامل على مقارنة العامل الأول إلى"نموذجpattern " نصي،والذي هو مجموعة من الحروف المتطابقة والقيم الشاملة wildcards .

```
If (someValue Like somePattern) Then
```

ليس عليك استخدام نفس قواعد النموذج pattern عند إعادة تعريف المعامل Like وتستطيع أن تقبل أي نوع بيانات من أجل عامل النموذجpattern ، ولكن يجب عليك دائماً العودة بنتيجة منطقيةBoolean .

```
Public Shared Operator Like(ByVal operand1 As Bumblebee, ByVal operand2 As Integer) As Boolean
    ' ----- See if Bumblebee matches an Integer pattern.
End Operator
```

لايوجد معامل "مرافق buddy" والذي يجب عليك معاملته عند إعادة قيادة المعامل Like.

المعاملات المنطقية Logical و Bitwise على مستوى البت Bitwise and Logical Operators

توجد أربع معاملات من بين المعاملات المنطقية logical والتي على مستوى البت (الوحدة التخزينية)المضمنة في الفيجوال بيسك، تنجز واجب مضاعف كمعاملات معاد تعريفها.تقبل المعاملات على مستوى البت(وحدة تخزينية)(وهي And، Or،Xor و Not)عوامل عددية صحيحةinteger، وتولد نتائج عددية مع القيم المنقولة(المرسلة) على مستوى الوحدة التخزينية المستقلة.وهي تعمل أيضاً كمعاملات منطقية،تقبل وتعيد قيم منطقيةBoolean،على الأغلب في العبارات الشرطية.ولكن بإمكانها معالجة الضغط كونها قابلة لإعادة القيادة أكثر بقليل.

عندما تعمل على إعادة قيادة override هذه المعاملات الأربع،فأنت تعيد قيادة إصدارات على مستوى البت (الوحدة التخزينية)bitwise versions وليس الإصدارات المنطقيةlogical . بشكل أساسي،هذا يعني أن لديك تحكم على القيمة المعادة،وليس مطلوب جعلها منطقيةBoolean .

تجدول القائمة التالية ثمانية معاملات على مستوى الوحدة التخزينية والمنطقية القابلة لإعادة التعريف.

المعاملOperator	التعليقComments
<<	يعمل معامل الرفع اليساري shift left رفع وحدة تخزينية على قيمة عددية صحيحة مصدرية،نقل الوحدات التخزينية bits إلى اليسار بواسطة عدد معين من المواضع.على الرغم من أنه ليس عليك استخدام هذا المعامل لعمل رفع وحدة تخزينية صحيحة،عليك قبول كمية رفع ما(عددية صحيحةInteger) كعامل ثاني.

```
Public Shared Operator <<(ByVal operand1 As Bumblebee, ByVal operand2 As Integer) As Bumblebee
    ' ----- Add shifting code here.
End Operator
```

المعاملOperator	التعليقComments
>>	يعمل معامل الرفع اليميني shift right رفع وحدة تخزينية تماماً مثل معامل الرفع اليساري،ولكن تنقل الوحدات التخزينية في اتجاه "اليمين right ".إني أظن أن هذا سيجعل هذه الوحدات التخزينية أكثر محافظةconservative. يمكن لكودك أن يجعل القيمة المعادة أكثر تساهلاً إذا أردت،ولكن كما مع معامل الرفع اليساري،يتوجب عليك قبول عدد صحيح Integer كعامل ثاني.
Not	معامل نفي على مستوى الوحدة التخزينية وهو أحادي،يقبل فقط معامل نسبي كعامل مفرد فقط.
And	معامل ربط conjunction على مستوى الوحدة التخزينية يضع وحدة تخزينيةbit في القيمة المعادة إذا ما تم إعداد وحدتي تخزين على حد سواء متوضعتين في عوامل المصدر.
Or	يضع معامل الفصلdisjunction على مستوى الوحدة التخزينية بت(وحدة تخزينية)في القيمة المعادة إذا ما تم إعداد الوحدات التخزينية المتوضعة بالتساوي في عوامل المصدر.
Xor	معامل الاستثناء على مستوى البتbitwise exclusion operator، يضع بت في القيمة المعادة إذا وفقط إذا كان واحد من البتات المتوضعة بالتساوي في عوامل المصدر تم إعداده.
IsTrue	إعادة تعريف المعامل Or لايعمل على إعادة تعريف المعامل OrElse المتعلق به بشكل آلي.لاستخدام OrElse يتوجب عليك إعادة تعريف المعامل IsTrue الخاص.وهو ليس معامل فيجوال بيسك حقيقي،ولا تستطيع استدعاءه بشكل مباشر حتى عندما يتم إعادة تعريفه.ولكن عندما تستخدم المعامل OrElse مكان المعامل Or المعاد تعريفه،تستدعي الفيجوال بيسك المعامل IsTrue عند الحاجة.يوجد عدة قواعد عليك إتباعها لاستخدام إعادة تعريف IsTrue. المعامل Or المعاد تعريفه يجب أن يعود بنوع الفئة للفئة التي بها تم تعريفه.إذا أردت استخدام OrElse على الفئة Bumblebee،فإن إعادة تعريف المعامل Or في تلك الفئة يجب أن يعود بقيمة من نوع Bumblebee. المعامل IsTrue المعاد تعريفه يجب أن يقبل عامل مفرد من نوع الفئة المحتوية(Bumblebee)،ويعود بمنطقيةBoolean . يتوجب عليك إعادة تعريف المعامل IsFalse. كيف تحدد الصحة truth أو الزيف من Bumblebee يعود إليك.
IsFalse	إعادة تعريف IsFalse يعمل تماماً مثل IsTrue ولديه نفس القواعد،ولكن يقدم إلى المعاملات AndوAndAlso .

المعامل "تحويل النوع" The CType Operator

تبدو ميزة الفيجوال بيسك CTypeأكثر شبها لدالة منها لمعامل:

```
result = CType(source, Type)
```

ولكن المظاهر خداعةlooks are deceiving. فهي ليست دالة حقيقية،وكما مع دوال التحويل الأخرى(مثل CInt)،يتم معالجتها عملياً وقت الترجمة، قبل تشغيل البرنامج بوقت طويل.من خلال السماح لك بإعادة تعريفها كالمعامل،تمكنك الفيجوال بيسك من إنشاء تحويلات conversionsنوعية وخاصة بين أنواع البيانات التي لاتبدو منسجمةcompatible. قالب الطريقة التالية يحول قيمة من نوع Bumblebee إلى عددي صحيحInteger .

```
Public Shared Operator CType(ByVal operand1 As Bumblebee) As Integer
    ' ----- Perform conversion here, returning an Integer.
```

```
   End Operator
```

إذا حاولت كتابة المقطع الأخير من الكود ضمن الفيجوال بيسك،سيعترض complainأنك تفقد إما الكلمة المحجوزة Widening أو الكلمة المحجوزة Narrowing(شاهد الشكل التالي).

```
                    Conversion operators must be declared either 'Widening' or 'Narrowing'.
Public Shared Operator CType(ByVal operand1 As Bumblebee) As Integer
    ' ----- Perform conve[!]on here, returning an Integer.
End Operator
```

لقد ذكرت تحويلات التوسيع narrowing والتضييق widening في الفصل الثاني،ولكن لنختبرها هنا بعمق أكثر.عندما تحول بين بعض أنواع البيانات الجوهرية في الفيجوال بيسك ،يوجد فرصة لأن يفشل التحويل في بعض الأحيان لأن القيمة المصدرية لا تتناسب في القيمة المقصودة.وهذا صحيح عند تحويل قيمة لنوع قصير Short إلى بايت Byte .

```
Dim quiteBig As Short = 5000
    Dim quiteSmall As Byte
    ' سيفشل السطرين التاليين
    quiteSmall = quiteBig
    quiteSmall = CByte(quiteBig)
```

وإنه من الواضح لما يفشل التحويل:المتغير من النوع بايت لايمكن أن يحفظ قيمة 5000.ولكن ماذا حول هذا الكود؟

```
Dim quiteBig As Short = 5
    Dim quiteSmall As Byte
    ' ----- These next two lines will succeed.
    quiteSmall = quiteBig
    quiteSmall = CByte(quiteBig)
```

إنه يعمل بشكل جيد،بما أن 5 تتناسب ضمن المتغير بايت مع مساحة احتياطية.(إذا كان خيار التدقيق Option Strict تم وضعه إلى فعال On، فإن الإسناد الأول سيفشل في الترجمة)، مع ذلك،لايوجد شيء يوقفني عن إسناد قيمة 5000 إلى المتغير quiteBig وتجريب الإسناد مرةً أخرى.هذا احتمالي *potential* الفشل خلال التحويل وهذه هي القضية.

عندما يكون للتحويل احتمال الفشل تبعاً لكون البيانات المصدرية غير قادرة على أن تتناسب بالكامل في المتغير المستهدف،فإنه يدعى تحول التضييق *narrowing conversion*.تحويلات التضييق *narrowing conversion* هي حقيقة،وطالما أنك قد اختبرت البيانات قبل التحويل،فلن يكون هناك أي سبب لتقيد مثل هذه التحويلات بشكل ثابت.

تذهب تحويلات التوسيع *Widening conversions* في اتجاه معاكس.فهي تحدث عندما تتناسب أي قيمة مصدرية في نوع البيانات المصدري في النوع الهدف دائما.تحويلات التوسيع ستنجح دائماً طالما أن البيانات المصدرية محققة.

تسمح الفيجوال بيسك للتحويلات التوسيع أن تحدث بشكل آلي،وبشكل ضمني.ليس عليك بشكل صريح استخدام CType لإجبار التحويل.إذا كان لديك تحول توسيع من فئة Bumblebee إلى العددي الصحيح Integer، وقد وضعت خيار التدقيق Option Strict إلى فعال On، فإن الكود التالي سيعمل جيداً:

```
Dim sourceValue As New bumblebee
    Dim destValue As Integer = sourceValue
```

إذا كان التحويل من إلى تضييق،فسيكون عليك إجبار التحويل باستخدام CType وذلك للتأكيد على الفيجوال بيسك أنك تريد عمل هذا :

```
Dim sourceValue As New bumblebee
    Dim destValue As Integer = CType(sourceValue, Integer)
```

عندما تعمل على إنشاء تحويل مخصص مع المعامل CType المعاد تعريفه،يتوجب عليك إعلام الفيجوال فيما إذا التحويل هو توسيع أو تضييق وذلك بإدخال إما الكلمة المحجوزة Widening أو الكلمة المحجوزة Narrowing بين الكلمة المحجوزة Shared والكلمة المحجوزة Operator .

```
Public Shared Narrowing Operator CType(ByVal operand1 As bumblebee) As Integer
    ' ----- Perform narrowing conversion here.
  End Operator
```

قضايا إعادة تعريف معامل أخرى.Other Operator Overloading Issues

يوجد أيضاً العديد من القواعد التي يجب عليك إتباعها عند إعادة تعريف المعاملات،ولكن في البداية لنلقي نظرة على شبه الفئة Bumblebee المفيدة .

```
Class Bumblebee
  Public Bees As Integer
  Public Sub New()
    ' ----- Default constructor.
    Bees = 0
  End Sub
  Public Sub New(ByVal startingBees As Integer)
```

```
        ' ----- Assign an initial number of bees.
        Bees = startingBees
    End Sub
    Public Shared Operator +(ByVal operand1 As Bumblebee, ByVal operand2 As Bumblebee) As Bumblebee
        ' ----- Join bumblebee groups.
        Dim newGroup As New Bumblebee
        newGroup.Bees = operand1.Bees + operand2.Bees
        Return newGroup
    End Operator
    Public Shared Operator -(ByVal operand1 As Bumblebee, ByVal operand2 As Bumblebee) As Bumblebee
        ' ----- Separate bumblebee groups.
        Dim newGroup As New Bumblebee
        newGroup.Bees = operand1.Bees - operand2.Bees
        If (newGroup.Bees < 0) Then newGroup.Bees = 0
        Return newGroup
    End Operator
    Public Shared Operator *(ByVal operand1 As Bumblebee, ByVal operand2 As Bumblebee) As Bumblebee
        ' ----- Create a swarm.
        Dim newGroup As New Bumblebee
        newGroup.Bees = operand1.Bees * operand2.Bees
        Return newGroup
    End Operator
    Public Shared Widening Operator CType(ByVal operand1 As Bumblebee) As Integer
        ' ----- Perform conversion here.
        Return operand1.Bees
    End Operator
End Class
```

الفئة بسيطة لحد ما،فهي موجودة لحفظ عدد بسيط من النحلات.ولكن بإعادة تعريف المعاملات الإضافة،الطرح،الضرب ،و"تحويل النوع"،نستطيع استخدام حالات من النحلات مع بناء أكثر واقعية.

```
Dim quiteBig As Short = 5
    Dim studyGroup1 As New Bumblebee(20)
    Dim studyGroup2 As New Bumblebee(15)
    Dim swarmGroup As Bumblebee = studyGroup1 * studyGroup2
    MsgBox("The swarm contains " & CInt(swarmGroup) & " bees.")
```

شغل الكود بشكل صحيح يولد الرسالة المبينة في الشكل التالي:



تضمين إعادة تعريف CTypeوالتي تولد عدد صحيح Integerيسمح لي تحويل Bumblebeeباستخدام المعامل CInt. وأستطيع أيضاً تغير السطر الأخير لاستخدام المعامل CTypeالصحيح.

```
MsgBox("The swarm contains " & CType(swarmGroup, Integer) & " bees.")
```

متطلبات التصريح Declaration Requirements

كما ذكرت سابقاً،يتوجب عليك جعل طرق المعامل عامة ومتشاركةPublic Shared. ولأن المعاملات المعاد تعريفها تحتاج نوعاً ما اتصال حميم بالفئة الحاوية عليهم،على الأقل واحد من العوامل أو القيمة المعادة يجب أن تطابق نوع الفئة المحتوية.(في بعض إعادة التعريف،تتطلب الفيجوال بيسك أن يكون واحد من المعاملات متطابقة)أي من إعادة التعريف التالي سيعمل تماماً وبشكل جيد،بما أن الـ Bumblebeeيتم استخدامها لواحد من المعاملات:

```
Public Shared Operator <=(ByVal operand1 As Bumblebee, ByVal operand2 As Integer) As Boolean
    ' ----- Compare a bumblebee to a value.
  End Operator
  Public Shared Operator <=(ByVal operand1 As Date, ByVal operand2 As Bumblebee) As Boolean
```

```
      ' ----- Compare a date to a bumblebee.
   End Operator
```

مهما يكن لاتستطيع وضع كلا العاملين إلى نوع غير النوع Bumblebeeفي نفس الوقت،وتبقى محتفظاً بإعادة التعريف في فئة Bumblebee.

```
Class Bumblebee
   Public Shared Operator <=(ByVal operand1 As Date,ByVal operand2 As Integer) As Boolean
      ' ----- هذا لن يتم ترجمته.
   End Operator
End Class
```

إعادة تعريف ما تم إعادة تعريفه.Overloading Overloads

تستطيع إعادة تعريف المعاملات المعاد تعريفها.فيمكنك إضافة تنوعات لعدة تواقيع للقيم المعادة والمعاملات النسبية لمعامل معاد تعريفه لفئة مفردة.

```
Public Shared Widening Operator CType(ByVal operand1 As Bumblebee) As Integer
      ' ----- اعمل تحويل إلى عدد صحيح هنا.
   End Operator
   Public Shared Widening Operator CType(ByVal operand1 As Bumblebee) As Date
      ' ----- اعمل تحويل إلى تاريخ هنا.
   End Operator
```

طالما أن تواقيع المعامل النسبي argument signatures أو القيمة المعادة مختلف،تستطيع إضافة العديد من إعادة تعريف المعامل كما تشاء.لا تحتاج إلى استخدام الكلمة المحجوزة Overloadsأيضاً

طرق التوسيع(التمدد)Extension Methods

ماذا بشأن الرغبة في تعديل سلوك فئة ما،ولكن ليس لديك إمكانية الوصول إلى كودها المصدري؟تستطيع الاشتقاق منها وتبني فئة جديدة،ولكن هذا ليس مناسب دائماً.تستطيع الاتصال بالمطور الأصلي وترجاه من أجل الكود،ولكن بعض هؤلاء المبرمجين بخيلين جداً tight-fisted عندما يتعلق الأمر ببرمجياتهم.

يوجد خيار أخر وهو استخدام ميزة الفيجوال بيسك2008 الجديدة :وهي طرق التوسيع(التمديد)*extension methods*. إليك كيفية العمل:

1.إنك تقرر أي فئة تريد توسيعها بالطرق الجديدة.

2.تكتب هذه الطرق ضمن وحدة برمجية Moduleقياسية في كودك المصدري.

3.تبدأ باستخدام الطرق الجديدة وكأنه تم تضمينها في تعريف الفئة.

يتضمن نوع بيانات السلسلة الحرفية String (النصية) العديد من الطرق الجاهزة(المبنية داخلياً) والتي تعود بنسخة معدلة لنسخة سلسلة حرفية.على سبيل المثال، في هذا الكود:

```
Dim bossyString As String = "are you talking to me?"
      MsgBox(bossyString.ToUpper())
```

النص الذي يظهر في صندوق الرسالة سيكون كله في الحالة الكبيرة uppercaseلأن الطريقة ToUpper تعود بنسخة حالة كبيرة جديدة من حالة (نسخة) النص الأصلي. الطريقة المطابقة ToLowerتعمل بطريقة أخرى(تحول الأحرف إلى الحالة الصغيرة)،ولكن ما أريده بالفعل هو الطريقة "إلى عنوان ToTitle "والتي تكبر فقط الحرف الأول لكل كلمة.

```
MsgBox(bossyString.ToTitle( ))
```

ولكن فئة السلسلة الحرفية لا تتضمن الطريقة "ToTitle "،ولكن نستطيع إضافتها بفضل طرق التوسيع(التمدد).لإنشاء طريقة تمدد،أنشئ طريقة ضمن وحدة برمجية moduleقياسية بحيث تقبل نوع البيانات المستهدفة كوسيط أول لها.

```
Module MyExtensions
   <System.Runtime.CompilerServices.Extension()> Public Function ToTitle(ByVal sourceText As String) As String
      Return StrConv(sourceText, VbStrConv.ProperCase)
   End Function
End Module
```

بالعادة،ستستدعي هذه الدالة كتمرير في السلسلة الحرفية الأصلية

```
MsgBox(ToTitle(bossyString))
```

وذلك الكود سيعمل تماماً،ولكن الزيادة في مواصفة التوسيع(من فضاء الأسماءSystem.Runtime.CompilerServices)تحول ToTitleإلى طريقة توسيع،توسع نوع البيانات النصية.إن كودك في الحقيقة لا يعدل نص.خلف الكواليس، يعمل مترجم الفيجوال بيسك على تحويل بناء الطريقة المشابهة الجديدة new method-like syntax إلى بناء الدالة المشابهة القديمة old function-like syntaxفي كل استخدام لـ ToTitle.

بنفسها طرق التوسيع لاتعمل الكثير.استدعاء ToTitle(bossyString) لايختلف عن bossyString.ToTitle( ).ولكن كما مع العديد من الميزات الجديدة في الفيجوال بيسك2008،طرق التوسيع تم إضافتها فقط لرفع raiseسعر المنتج.إني أمزح،إن ميزة طرق التوسيع الجديدة دعم هام من أجل تقنية لينكو LINQالجديدة.

مشروع.Project

سنضيف الكثير من الكود في مشروع هذا الفصل إلى تطبيق المكتبة،أكثر من 25%من أساس الكود الكامل.معظمه مشابه للكود الذي أضفناه في الفصول السابقة،يوجد الكثير للقراءة هنا،ولكن كلما أضفت فورم جديد للمشروع تأكد من إلقاء نظرة على كودها لتصبح على إطلاع بعملها الداخلي.

إعادة تعريف تحويلOverloading a Conversion

إن إعادة تعريف معامل أداة مفيدة،وأنا مولع خاصة بإعادة تعريفCType .لنعمل على إضافة إعادة تعريف CType إلى الفئة ListItemData التي صممناها في الفصل الثامن. ،تعرض هذه الفئة كل من الخاصيات ItemTextوItemData، موفرةً إمكانية الوصول إلى السمات النصية textualوالعددية numericلمحتوى الفئة.وإن هدفها الرئيسي دعم تتبع أرقام IDفي أدوات الصندوق المركب وصندوق القائمة.إذا كنا بحاجة لمعرفة رقم IDلبند مختار في أداة صندوق قائمة(لنسمه SomeList)،فإننا نستخدم كود مشابه للتالي:

```
Dim recordID As Integer = CType(SomeList.SelectedItem, ListItemData).ItemData
```

لايوجد شيء خاطئ في هذا الكود.ولكن إني أعتقد،"أليس من الجميل تحويل حالة إلى انتغر باستخدام الدالةCInt، وليس علينا التورط بمتغيرات عضو مثل ItemData ؟"

```
Dim recordID As Integer = CInt(CType(SomeList.SelectedItem, ListItemData))
```

الكود غير مختلف.ولكن لدعم هذا التحويل،نحتاج إلى إضافة إعادة تعريف CTypeإلى فئة ListItemData.افتح ملف تلك الفئة، وأضف الكود التالي كعضو للفئة.

```
  Public Shared Widening Operator CType(ByVal sourceItem As ListItemData) As Integer
' للتحويل إلى انتغر,ببساطة استخرج عنصر انتغر
        Return sourceItem.ItemData
    End Operator
```

هذا بسيط جداً،هذا التحويل الموسع من ListItemDataإلى انتغر يعمل على إعادة قسم الانتغر من الحالة.يوجد حوالي أربع أو خمس أمكنة في مشروع المكتبة الحالي بإمكانها الوصول مباشرةً إلى العضوItemData، وهي ليست بتلك الأهمية للعودة وتغيرها.ولكن سنستخدم إعادة تعريف التحويل بشكل متكرر في الكود الجديد المضاف في هذا الفصل.

ميزات الدعم الشاملة.Global Support Features

نحتاج إلى إضافة العديد من المتغيرات والروتينات الشاملة لدعم الميزات المتنوعة المستخدمة على طول التطبيق.سوف يتتبع متغيران شاملان الإعدادات المخزنة في جدول SystemValue لقاعدة البيانات.أضفهم كأعضاء إلى الوحدة البرمجية General (في *General.vb*)

```
Public DefaultItemLocation As Integer
Public SearchMatchLimit As Integer
```

يطابق برنامج المكتبة الكتب والبنود الأخرى كما تم تخزينها في المواضع المتعددة مثل التشعبات(الفروع) المتعددة أو مساحات التخزين.يشير DefaultItemLocation إلى أي واحد من هذه المواضع،من جدول CodeLocation،وهو الافتراضي.تخزن مدخلة DefaultLocation لجدول قاعدة البيانات SystemValue هذه القيمة بشكل دائم.عند البحث عن كتب،مؤلفين،أو أشياء أخرى يمكن أن تنتج آلاف المتطابقات،يدلSearchMatchLimit على العدد الأكبر من المطابقات المعادة بعمليات بحث مثل هذه.ويتم تخزينها كقيم نظام محددة البحثSearchLimit .

بما أننا حالياً في الوحدة البرمجيةGeneral، أضف دالتي مساعدة إضافيتين:

```
Public Function ConfirmDefaultLocation() As Boolean
' العودة بصواب إذا تم تحديد الموضع الافتراضي.
        Dim locationID As String
' الحصول على الموضع الافتراضي عند الحاجة.
        If (DefaultItemLocation = -1) Then
            locationID = GetSystemValue("DefaultLocation")
            If (locationID = "") Then locationID = "-1"
            DefaultItemLocation = CInt(locationID)
        End If
' العودة بنتيجة الحالة.
        If (DefaultItemLocation = -1) Then
            MsgBox("You must first define the default item location.", _
                MsgBoxStyle.OKOnly Or MsgBoxStyle.Exclamation, ProgramTitle)
            Return False
        Else
            Return True
        End If
    End Function
Public Function GetCopyDisposition(ByVal copyID As Integer) As String
        ' ----- Given the ID of an item copy, return it's current status.
        '       The available results are:
        '         New Item Copy       Error or missing record
        '         Checked In          Copy is checked in and available
        '         Checked Out         Copy is checked out by a patron
        '         Overdue             Copy is checked out and late
        '         Missing             Copy has been reported missing
        '         Reference           Copy is a reference copy
        Dim sqlText As String
        Dim dbInfo As SqlClient.SqlDataReader
        Dim response As String
        On Error GoTo ErrorHandler
        ' ----- Retrieve information about the book.
        sqlText = "SELECT IC.Reference, IC.Missing, PC.Missing AS PatronMissing, " & _
            "PC.DueDate, PC.Patron FROM ItemCopy AS IC LEFT JOIN PatronCopy AS PC " & _
            "ON IC.ID = PC.ItemCopy WHERE IC.ID = " & copyID & _
            " AND PC.Returned = 0"
```

```
        dbInfo = CreateReader(sqlText)
        If (dbInfo.Read = False) Then
            ' ----- Perhaps the database, due to the JOIN clause, could not return
            '       the base record of a checked-in book.
            dbInfo.Close()
            sqlText = "SELECT Reference, Missing, NULL AS PatronMissing, " & _
                "NULL AS DueDate, NULL AS Patron FROM ItemCopy WHERE ID = " & copyID
            dbInfo = CreateReader(sqlText)
            If (dbInfo.Read = False) Then
                ' ----- What happened to the record?
                dbInfo.Close()
                dbInfo = Nothing
                Return "New Item Copy"
            End If
        End If
        ' ----- We found the record. Check its details.
        If (IsDBNull(dbInfo!Patron) = False) Then
            ' ----- This item is checked out.
            If (CBool(dbInfo!Missing) = True) Or (CBool(dbInfo!PatronMissing) = True) Then
                response = "Missing"
            ElseIf (CDate(dbInfo!DueDate) < Today) Then
                response = "Overdue"
            Else
                response = "Checked Out"
            End If
        Else
            ' ----- This book is currently checked in.
            If (CBool(dbInfo!Missing) = True) Then
                response = "Missing"
            ElseIf (CBool(dbInfo!Reference) = True) Then
                response = "Reference"
            Else
                response = "Checked In"
            End If
        End If
        dbInfo.Close()
        dbInfo = Nothing
        Return response
ErrorHandler:
        GeneralError("GetCopyDisposition", Err.GetException())
        On Error Resume Next
        If Not (dbInfo Is Nothing) Then dbInfo.Close() : dbInfo = Nothing
        Return "New Item Copy"
    End Function
```

ConfirmDefaultLocation

يتحقق هذا الروتين من وجود مدخلة الموضع الافتراضي DefaultLocation في الجدول SystemValue.ويعمل على إرجاع صواب في حالة النجاح.

GetCopyDisposition

يوفر هذا الروتين وصف قصير للحالة الحالية من نسخة بند مكتبة معين.إنه يحلل سجلات العميل والبند،ويعيد واحد من حالات نصوص الكود:نسخة بند جديد New Item Copy،المدخلات Checked In، المخرجات Checked Out، متأخرات Overdue، مفقود Missing، أو دليل(مرجع Reference).

توسيع فئة مزودة من قبل إطار العمل.Extending a Framework-Supplied Class

ماذا يوجد في الاسم؟،حسناً،إذا كان اسم المؤلف author في مشروع المكتبة،يمكن أن يتضمن الاسم الأول والاسم الأخير first and last names، السابقات prefixes مثل("د" Dr)ولاحقات suffixes مثل("القروي")، وتواريخ dates من أجل الميلاد والموت birth and death. بعض من هذه الأجزاء اختيارية، لذلك تنسيق اسم المؤلف عملية متعددة الخطوات.بما أن التطبيق سيحتاج إلى تنسيق أسماء المؤلفين في عدة أماكن على طول الكود،دعنا نضيف روتين مركزي، FormatAuthorName والذي يقوم بهذا العمل.

```
<System.Runtime.CompilerServices.Extension()> Public Function FormatAuthorName(ByRef dbInfo As
SqlClient.SqlDataReader) As String
        ' ----- Given an author record, return the formatted name.
        Dim authorName As String
        On Error Resume Next
        ' ----- Format the name.
        authorName = CStr(dbInfo!LastName)
        If (IsDBNull(dbInfo!FirstName) = False) Then
            authorName &= ", " & CStr(dbInfo!FirstName)
            If (IsDBNull(dbInfo!MiddleName) = False) Then _
                authorName &= " " & CStr(dbInfo!MiddleName)
        End If
```

```
        If (IsDBNull(dbInfo!Suffix) = False) Then _
            authorName &= ", " & CStr(dbInfo!Suffix)
        ' ----- Add in the birth and death years.
        If (IsDBNull(dbInfo!BirthYear) = False) Or _
                (IsDBNull(dbInfo!DeathYear) = False) Then
            authorName &= " ("
            If (IsDBNull(dbInfo!BirthYear) = True) Then
                authorName &= "????"
            Else
                authorName &= CStr(Math.Abs(CInt(dbInfo!BirthYear)))
                If (CInt(dbInfo!BirthYear) < 0) Then authorName &= "BC"
            End If
            authorName &= "-"
            If (IsDBNull(dbInfo!DeathYear) = False) Then
                authorName &= CStr(Math.Abs(CInt(dbInfo!DeathYear)))
                If (CInt(dbInfo!DeathYear) < 0) Then authorName &= "BC"
            End If
            authorName &= ")"
        End If
        ' ----- Finished.
        Return authorName
    End Function
```

هذا الروتين هو طريقة توسيع تعمل على تمديد الفئة SqlClient.SqlDataReader.تصنع المواصفة Extensionالمؤهلة بالكامل الاتصال بين التوسع الخاص بنا وفئة إطار العمل المحددةSqlDataReader . منح SqlDataReader المبنية من سجلات في جدول Author،ودالة التنسيق والعودة باسم المؤلف بشكل لائق في التنسيق "Public, John Q, Jr. (1900–1999)." في مكان أخر من التطبيق،يتم استدعاء هذا الروتين وكأنه عضو من حالة قارئ بيانات:

```
dbInfo.FormatAuthorName( )
```

نستطيع أن نهمل ميزات طريقة التمديد بالكامل بحذف مواصفة التمديد Extension attribute بكل بساطة.ومن ثم،سيبدو استدعاء تنسيق المؤلف كالتالي:

```
FormatAuthorName(dbInfo)
```

محررات السجلات ودعم النماذج.Record Editors and Supporting Forms

الآن بدأت الأشياء بالوثوب.سنعمل على إضافة 23فورم إلى التطبيق في هذا الفصل.معظمها تنفذ محررات الكود القاعدي،مشابهة لملفات *UserName.vb* و *GroupName.vb* التي بنيناها في الفصل 11. نماذج أخرى موجودة لتوفير دعم إضافي لمحررات السجلات هذه.

نماذج تحديد البحثSearch-limiting forms

تسمح النماذج الأربع الأولى لأمين المكتبة تحديد إعادة تعريف(تحميل)المعلومات التي تأتي من خلال استخدام قاعدة البيانات مع الآلاف من الكتبbooks، المؤلفين authors،والناشرينpublishers . من المحتمل أنك تتذكر أن النموذج ListEditRecordsالشامل يعرض جميع السجلات الموجودة من سجلات جدول بشكل افتراضي.وهذا يعمل بشكل جيد من أجل مجموعات الأمن security groups المخزنة في جدول GroupNameإلا أنه ومن المحتمل ليس لديك الكثير منها.ولكن جدولة جميع الكتب في مكتبة حتى ولو كانت صغيرة يمكن أن يولد قائمة مفروضة تماماً.وبالاعتماد على سرعة كمبيوترك،فيمكن أن يأخذ فترة لتحميل عناوين الكتب ضمن القائمة.

تساعد النماذج الأربع"تحديد-البحث"على تخفيض عدد السجلات التي تظهر في القائمة في الحال.عندما يتمكن أمين المكتبة librarianمن الوصول إلى قائمة الكتب وبنود المكتبة الأخرى،فإن النموذج ItemLimitيوفر حث(طلب)بحث سريع والذي يقلل النتائج المجدولة.



تتيح هذه الفورم للمستخدم استخلاص جميع السجلات،أو بنود معينة بالاعتماد على اسم بند(مع دعم القيمة الشاملة).حالما يتم تحميل المتطابقات،يمكن للمستخدم الوصول إلى هذه الفورم مرةً أخرى بالنقر على "بحث Lookup " على النموذج ListEditRecordsمن أجل هذه الأنواع من محررات الكود التي تدعم البحث عن(المؤلفين،بنود،زبائن،والناشرين).نحن الآن جاهزين لتضمين نماذج تحديد البحث الأربعة في المشروع:

AuthorLimit.vb

تحدد الفورم سجلات المؤلف كما تم تحميلها من جدول المؤلفAuthor .

ItemLimit.vb

هذه هي الفورم التي تحدثنا عنها سابقاً.فهي تحدد عرض بنود المكتبة من جدول NamedItem.

PatronLimit.vb

في حال احتشاد الزبائن على مكتبتك،تتيح لك هذه الفورم تحديد السجلات المحملة من جدول الزبون Patron.

PublisherLimit.vb

تحدد هذه الفورم السجلات من جدول الناشرين Publisher.

محررات الموضوع والكلمات المفتاحية.Keyword and subject editors

على الرغم من أن معظم محررات السجلات توفر تجربة تحرير كاملة من خلال الفورم ListEditRecords،بعضها خاضع لنماذج تحرير أخرى.فالكلمات المفتاحية Keywordsوالمواضيع subjectsهي مثال جيد.حيث أن لكل منها جداوله الخاصة(الكلمة المفتاحية والموضوع).اخترت السماح بتحريرها من خلال النموذج الذي يحرر بنود المكتبة المستقلة، أي النموذج NamedItem (سنضيفه فيما بعد).تدير تلك الفورم جميع التفاعلات بين سجلات الموضوع والكلمة المفتاحية وجدول NamedItem، كل هذا من خلال الجداول الوسيطة ItemKeyword وItemSubject لعلاقة عديد إلى عديد.توفر النموذج KeywordAdd والنموذجSubjectAdd مدخلة نص بسيط من أجل كلمة مفتاحية أو موضوع.اعمل على تضمين كل هذه النماذج الآن في المشروع:

- النموذج إضافة كلمة مفتاحية.*KeywordAdd.vb*
- النموذج إضافة موضوع.*SubjectAdd.vb*

نماذج إضافية لدعم البنود المسماة.More named item support forms

كما سنرى فيما بعد،إن نموذج NamedItemواحد من النماذج الأكثر تعقيداً المضافة إلى مشروع المكتبة حتى الآن. فهو يدير كل شيء يخص بند مكتبة مولد(مثل كتاب ما).كل بند يمكن أن يكون لديه عدة نسخ،مؤلفين،كلمات مفتاحية،مواضيع،وهكذا.من أجل التحرير الفعال فمن البساطة تضمينها جميعا على نموذج مفرد.عملنا سابقاً على إضافة نموذجين تابعين(أو ثانويين): KeywordAdd وSubjectAdd.لدعنا نعمل على إضافة خمس نماذج دعم إضافية.

AuthorAddLocate.vb

تمثل هذه الفورم واجهة مشابهة لمعالج سحري يتيح للمستخدم إضافة سجل مؤلف جديد أو موجود لبند.والمؤلفين في برنامج المكتبة مصطلح شامل يشير إلى المؤلفين authors،المحررين editors، المفسرين illustrators، الممثلين أو الفنانين performers، وهكذا،خطوات المعالج السحري الثلاث للنموذج تسمح للمستخدم(1)الإشارة إلى نوع المؤلف بواسطة جدول CodeAuthorType،(2) عمل بحث عن مؤلف موجود بالاسم،و(3)اختيار أسماء مؤلفين من قائمة مطابقة.إذا لم يكن المؤلف المطلوب في قاعدة البيانات حتى الآن،تسمح الخطوة الأخيرة بإضافة مؤلف جديد.يبين الشكل التالي أو خطوتين من الخطوات الثلاث.



يتم التحكم بمعظم المنطق من خلال معالج حدث زر التاليNext. تشكيلة الكود في هذا الروتين تعتمد على لوحة المعالج الحالي التي هي قيد العرض(كما تم الإشارة إليها من قبل المتغير ActivePanelالذي على مستوى الفئة).إليك الكود الذي يعمل عندما ينقر المستخدم على زر التالي بعد اختيار نوع المؤلفauthor type :

```
If (CInt(CType(NameType.SelectedItem,ListItemData)) = -1) Then
            MsgBox("من فضلك اختر نوع الاسم من القائمة",MsgBoxStyle.OkOnly Or
MsgBoxStyle.Exclamation,ProgramTitle)
            NameType.Focus()
            Return
        End If
        ' الانتقال إلى لوحة البحث
        ActivePanel = PanelCriteria
        SecondPanel.Visible = True
        FirstPanel.Visible = False
        ActBack.Enabled = True
        LastName.Focus()
```

ألا ترى سطر المنطق الأول في ذلك الكود؟إننا نستخدم دالة التحويل CIntللحصول على قيمة ItemDataمن بند قائمة.وهو استدعاء لمعامل CType الذي عملنا على إعادة تعريفه في فئة ListItemData.

PublisherAddLocate.vb

تشبه هذه الفورم تماماً الفورم السابقة،ولكن تركز على الناشرين.ومعالجها السحري لديه خطوتين فقط بما أن الناشرين غير مجمعين بواسطة النوعtype. إنها تعمل على إيجاد أو إضافة سجلات في جدول الناشرPublisher. عندما يحين الوقت لإضافة ناشر إلى بند ما،يستدعي فورم محرر البند الدالة PublisherAddLocate.PromptUser.وتعود هذه الدالة بالمعرف ID لسجل الناشر المختار،أو 1- لإجهاض إضافة ناشر.القيمة الراجعة-2 تعمل على تنظيف أي اختيار سابق.

SeriesAddLocate.vb

تشابه هذه الفورم الفورمPublisherAddLocate، ولكن تتبّق من أجل السجلات من جدول CodeSeries.

ItemAuthorEdit.vb

حالما يتم إضافة مؤلف إلى بند ما،الطريقة الوحيدة لتغيره إلى مؤلف مختلف هي إزالة المؤلف الغير صحيح،وإضافة المؤلف الصحيح بشكل منفصل من خلال الفورم AuthorAddLocate.ولكن إذا اختار المستخدم ببساطة نوع مؤلف خطأ(مثل "محررEditor " بدل "مفسر أو شارحIllustrator ")،فمن العبء البحث مرةً أخرى عن اسم المؤلف لتغير النوع فقط .تتيح الفورم ItemAuthorEditللمستخدم من تعديل نوع المؤلف المضاف سابقاً لبند ما.فهي تعدل حقل قاعدة البيانات ItemAuthor.AuthorType.

ItemCopy.vb

من المحتمل أن يكون لدى المكتبة عدة نسخ من كتاب معين،أو سيدي،أو أي بند أخر.في مشروع المكتبة،هذا يعني أن كل سجل NamedItemيمكن أن يكون لديه أكثر من سجل ItemCopyمرافق له .كل نسخة يتم تحريرها من خلال الفورم (شاهد الشكل التالي).



على الرغم من أن هذا الكود غير مشتق من BaseCodeFormكما مع المحررات الأخرى،فهو ما يزال لديه العديد من ميزات تلك النماذج،من ضمنها الروتين SaveFormDataالذي يكتب السجلات إلى قاعدة البيانات.

واحد من الأشياء المهمة التي تعملها هذه الفورم هو أن لديها دعم لقراءة الأكواد القابلة للقراءة من قبل الألة.تتصرف العديد من قارئات أكواد الآلة كمقحم wedge إدخال نص كود الآلة للبحث ضمن جدول إدخال لوحة مفاتيح keyboard input stream الكمبيوتر.فأي برنامج يعرض كود الآلة عليه عرض إدخال نص طبيعي. ماسحات حشر أكواد الآلة تزييل(أو تعمل على إلحاق) عودة المشيرة(المفتاح إنترEnter)إلى نهاية أكواد الآلة المنقولة.يسمح هذا للبرنامج من اكتشاف نهاية رقم كود الآلة.ولكن في معظم نماذج برنامج المكتبة،المفتاح انتر Enterيعمل على إطلاق الزر "موافق OK " ويغلق الفورم.نحن لا نريد أن يحدث هذا هنا.لمنع هذا،سنعمل على إضافة بعض الكود لهذه الفورم ليعمل على تعطيل النقر الآلي على الزر "موافق OK " كلما كانت نقطة الإدخال في حقل إدخال نص كود الآلة.

```
Private Sub RecordBarcode_Enter(ByVal sender As Object, ByVal e As System.EventArgs) _
Handles RecordBarcode.Enter
        ' تعليم كامل النص
        RecordBarcode.SelectAll()
        ' لاتسمح بانتر باغلاق الفورم
        Me.AcceptButton = Nothing
    End Sub
    Private Sub RecordBarcode_Leave( _
    ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles RecordBarcode.Leave
        ' اسمح لانتر باغلاق الفورم مرة أخرى
        Me.AcceptButton = ActOK
    End Sub
    Private Sub RecordBarcode_KeyPress(ByVal sender As Object, _
    ByVal e As System.Windows.Forms.KeyPressEventArgs) _
    Handles RecordBarcode.KeyPress
        ' تجاهل مفتاح انتر
        If (e.KeyChar = ChrW(Keys.Return)) Then e.Handled = True
    End Sub
```

مع هذا الكود ،عندما يضغط المستخدم على المفتاح انتر في حقل كود الآلة بشكل يدوي،فإن الفورم لن يتم إغلاقها.ولكنه ثمن بسيط تدفعه من أجل دعم كود الآلة.

محررات الكود المشتق.Inherited code editors

12 من النماذج المضافة في هذا الفصل تُشتق مباشرة من فئة الفورمBaseCodeForm. أضفها إلى المشروع كما سأرلجع كل منها:

Author.vb

يحدث نموذج المؤلف Authorالسجلات في جدول قاعدة البيانات Author. وكفئة مشتقة من الفورمBaseCodeForm. فإنها تعمل على إعادة قيادة العديد من العناصر العامة لفئتها القاعدية.اثنان من إعادة القيادة لم نعمل على استخدامهما من قبل في الفصول السابقة وهما الطريقة UsesSearchوSearchForRecord.تسمح هذه الطرق لمستخدم نموذج ListEditRecords تحديد المؤلفين المعروضين من خلال انبثاق النموذج AuthorLimitالذي تم شرحه سابقاً في هذا الفصل.(إعادة قيادة FillListWithRecordsيستدعي أيضاً SearchForRecordللطلب من المستخدم من أجل القائمة الأولية من المؤلفين لكي يتم عرضها).في SearchForLimit،استدعاء إلى الطريقة AuthorLimit.PromptUser يعمل على إرجاع نص مفصول بفواصل في تنسيق "الأخيرLast، الأولFirst ."

```
' الطلب من المستخدم تحديد اسم المؤلف
        exceededMatches = False
        userLimit = (New AuthorLimit).PromptUser()
        If (userLimit = "") Then Return
```

يمكن للمستخدم تضمين رمز النجمة(*) كقيمة شاملة wildcardفي بداية أو نهاية أجزاء الاسم.لقد أصبحت علامة النجمة حرف عام للاستخدام في جميع أنواع بحث القيمة الشاملةwildcard. لسوء الحظ،إنها غير مدعومة في عبارات السكول سيرفر "اختر SELECT." يستخدم السكول سرفر رمز النسبة المئوية(%) من أجل القيمة الشاملة (كما تعمل العديد من منصات قاعدة البيانات الخاضعة لسكول)بما أن SearchForLimitيستخرج الأسم الأول والأخير،فهو يضمن استخدام حرف شامل مناسب.

```
' استخدم التحديدات للمساعدة في تحضير النص
        limitLast = Trim(GetSubStr(userLimit, ",", 1))
        limitFirst = Trim(GetSubStr(userLimit, ",", 2))
        If ((limitLast & limitFirst) = "") Then Return
        If (InStr(limitLast, "*") = 0) Then limitLast &= "*"
        If (InStr(limitFirst, "*") = 0) Then limitFirst &= "*"
        limitLast = Replace(limitLast, "*", "%")
        limitFirst = Replace(limitFirst, "*", "%")
```

هذا الكود يستخدم روتيننا الخاص GetSubStrالذي أضفناه سابقاً إلى الوحدة البرمجية General.حالما يتم استخراج أجزاء الاسم،تعمل دالة الفيجوال بيسك Replaceعلى استبدال جميع نسخ * بـ %.ستجد كود مشابه في محررات السجلات الأخرى التي تسمح بتحديدات على قائمة السجلات.مثل نموذج المؤلف Publisherالذي سنضيفه فيما بعد.

بينما يكون الكود المصدري لهذه الفورم مفتوح انتقل للأعلى.ستجد هناك عبارة Imports الهامة.

```
Imports MVB = Microsoft.VisualBasic
```

عادة تكون عبارة Importsمتبوعة مباشرةً بفضاء أسماء.هذا التنوع يتضمن السابقة= MVB ،والتي تعرف مختصر لفضاء الأسماء Microsoft.VisualBasic من أجل الكود في هذا الملف.مع إحضار الفيجوال بيسك العديد من فضاءات الأسماء ضمن فئة موجودة تعرف أيضاً الكثير من الأعضاء العامة،يوجد حصر(تقييد)لأن تحصل تضاربات في اسم عضو.في هذه الحالة،التضارب هو عضو الفورم Left أو RIGHT.بما أن هذا الكود المصدري لنموذج المؤلف Authorيرى كل شيء من خلال منشور تلك الفورم،عندما تضمن الكلمة المحجوزة Leftفي منطقك،فإن الكود عادة يفترض أنك تعني خاصية الفورمLeft ،والتي تعمل على وضع الموقع اليساري للفورم.المشكلة أن Left هي دالة معالجة نصوص مشتركة تستخرج حروف الجهة اليسارية من سلسلة حرفية أكبر.

```
smallerString = Left(largerString, 5)
```

في نموذج ما،هذا الكود يولد خطأ بما أن الاعتقاد أن المقصود منه Me.Left.لاستخدام نسخة السلسلة الحرفية لـ Left،عليك اسباقها بفضاء أسمائها.

```
smallerString = Microsoft.VisualBasic.Left( largerString, 5)
```

تتيح لك عبارة Importsالخاصة استخدام اسم أقصر بديل من فضاء الاسماء Microsoft.VisualBasic الطويل نوعاً ما.

```
smallerString = MVB.Left(largerString, 5)
```

ستجد عدة حالات مشابه لهذا الكود في نماذج أخرى تتضمن الاختصار MVB.

نموذج المؤلف لديها عنصر أكثر أهمية،تظهر لصاقة مطابقة الاسم Name Matches قرب الحافة السفلية للفورم.كما هو مبين في الشكل التالي:



يساعد هذا الحقل المستخدم على تجنب إضافة نفس المؤلف إلى قاعدة البيانات مرتين.فعندما يتم عمل تغيرات على حقول الاسم الاخير والأول،يطابق الاسم حقل يتم تحديثه بأسماء المؤلفين المتطابقة الموجودة في جدول المؤلفAuthor. يحصي الروتين RefreshMatchingAuthorsعدد المؤلفين المتطابقة من خلال الكود التالي:

```
sqlText = "SELECT COUNT(*) AS TheCount " & "FROM Author WHERE LastName LIKE " &
DBText(Trim(RecordLastName.Text))
        If (Trim(RecordFirstName.Text) <> "") Then
       sqlText &= " AND FirstName LIKE " & DBText(MVB.Left(Trim(RecordFirstName.Text), 1) & "%")
        End If
        matchCount = CInt(ExecuteSQLReturn(sqlText))
```

وهذا مشابه لكود البحث في الروتينSearchForLimit، ولكنه يضيف فقط قيمة شاملة wildcardإلى الاسم الأول قبل عمل البحث.

CodeAuthorType.vb

تحرر هذه الفورم السجلات المتعلقة بجدول CodeAuthorType.

CodeCopyStatus.vb

تحرر هذه الفورم السجلات في جدول قاعدة البيانات CodeCopyStatus.

CodeLocation.vb

كما هو متوقع،تحرر هذه الفورم السجلات في جدول CodeLocation.حالما تعمل على إضافة سجل واحد على الأقل على ذلك الجدول،ستكون قادر على وضع الموضع الافتراضي لقاعدة البيانات.سأناقشه مرةً أخرى فيما بعد في هذا الفصل.

CodeMediaType.vb

تحرر هذه الفورم السجلات في الجدولCodeMediaType. متضمنة القليل من السجلات الإضافية أكثر من محررات جداول الكودCode. معظم الحقول تقبل إدخال عددي،على الرغم من أنني أعمل تفحص نهائي من أجل البيانات العددية المحققة(أو الصحيحة) تماماً قبل كتابة سجل إلى قاعدة البيانات،أحاول منع أي بيانات غير

عدية من الظهور في الموضع الاول وذلك بتقيد ضربات المفاتيح المقبولة.على سبيل المثال،حدث ضغط مفتاح حقل نص RecordCheckoutDays text field's يتضمن الكود التالي KeyPress:

```
'السماح للأرقام و الباك سبيس فقط
        If (e.KeyChar = vbBack) Or (sNumeric(e.KeyChar)) Then Return
        e.Handled = True
```

وضع الخاصية e.Handled إلى صواب True يوقف الفيجوال بيسك عن عمل أي شيء آخر مع المفتاح انتر.إنها طريقة سهلة وسريعة للتخلص من المفاتيح المدخلة من قبل المستخدم.

CodePatronGroup.vb

تحرر هذه الفورم السجلات في جدول CodePatronGroup.

CodeSeries.vb

هذا المحرر يدير السجلات في جدول CodeSeries. ذكرت سابقاً كيف يتم إتباع الكلمات المفتاحية keywords وأسماء السلاسل series names بنود مسماة named items.ولكن كان من الواضح لي أيضاً توفير إدارة مباشرة لأسماء السلاسل.في حال أردت تركيب قائمة مشتركة قبل إضافة بنود مكتوبة مستقلة.لذلك،تعمل هذه الفورم واجبين مضاعفين:تستطيع الوصول لها كمحرر سجلات قياسي من خلال الفورم ListEditRecords، ويتم استخدامها أيضاً من أجل بند مسمى بعينه من خلال نموذج NamedItem الغير مضافة حتى الآن.

عند تحرير أسماء سلسلة لبند معين،يعمل المستخدم أولاً بحث عن اسم السلسلة بكتابتها.بما أنني لا أريد من المستخدم إعادة كتابة اسم السلسلة مرةً أخرى في نموذج هذا المحرر،فأردت تمرير اسم السلسلة المكتوبة إلى نموذج CodeSeries،ولكن ولا واحد من الطرق المعاد قيادتها تدعم هذا.لذلك سنحتاج إلى إضافة طريقة جديدة تقبل الاسم المكتوب.العضو AddRecord الذي يعيد قيادة الدالة القاعدية لنفس الاسم .

```
Public Overrides Function AddRecord() As Integer
        ' إضافة سجل جديد
        ActiveID = -1
        PrepareFormFields()
        Me.ShowDialog()
        If (Me.DialogResult = Windows.Forms.DialogResult.OK) Then _
        Return ActiveID Else Return -1
    End Function
```

دعنا نضيف إعادة تعريف لهذه الدالة بحيث تتضمن معامل نسبي نصي.سيمرر المستدعي النص المكتوب الأصلي إلى هذا المعامل النسبي.سنعمل على إسناده إلى خاصية النص Text للأداة RecordFullName بحيث تظهر بشكل آلي عند فتح الفورم.

```
Public Overloads Function AddRecord(ByVal seriesText As String) As Integer
                ActiveID = -1
        PrepareFormFields()
        RecordFullName.Text = seriesText
        Me.ShowDialog()
        If (Me.DialogResult = Windows.Forms.DialogResult.OK) Then Return ActiveID Else Return -1
    End Function
```

كان بإمكاننا استخدام اسم ما بدل AddRecord لهذه الدالة وتجنب إضافة إعادة التعريف overload. ولكن من الجميل الحفاظ على الأشياء مستقرة(أو متطابقة).

Holiday.vb

تدير هذه الفورم السجلات في جدول .فيما بعد في فصل لاحق سنعمل على إضافة مجموعة من العطل ضمن البرنامج من أجل سرعة الوصول.

Patron.vb

نموذج الزبون توفر خدمات تحرير للسجلات في جدول الزبون Patron، وتظهر في الشكل التالي.



تتضمن هذه الفورم أداة تبويب TabControl للمساعدة على تقسيم عدد الحقول التي على المستخدم تجريبها في الحال.إذا كنت قد استخدمت أداة التبويب tab المضمنة مع الفيجوال بيسك6،ستقدر بسرعة استبدال الدوت نت.فهو يدير جميع منطق تبديل اللوحات بشكل آلي عندما يختار المستخدم تبويب مختلف.كل لوحة هي حالة فئة لصفحة تبويب TabPage منفصلة.في كودك،إجبار أداة التبويب على عرض تبويب مختلف سهل كإسناد حالة صفحة تبويب TabPage مناسبة إلى الخاصية SelectedTab للكائن TabControl.كما مع سطر الكود من الدالة ValidateFormData.

```
TabPatron.SelectedTab = TabGeneral
```

على الرغم من أن هذه الفورم قد تبدو معقدة تماماً،فهي مركبة بالكامل من معظم الكود الذين رأيناه في النماذج الأخرى.خلف إعادة القيادة لأعضاءBaseCodeForm ، تتضمن هذه الفورم دعم تفحص(أو مسح)كود التعريفbar code (كود تعريف الآلة على أداة ما)المستعارة من النموذجItemCopy، منطق كلمة السر password المسروقة من فورم UserName،والكود المطابق لاسم مشابه إلى ماهو مستخدم في نموذج Author.

عملت على تضمين زر إدارة بنود زبون Manage Patron's Itemsعلى الفورم،ولكن لن نعمل على إضافة منطقه حتى فصل متأخر.وعند ذلك الوقت ستصبح الدالة العامة الإضافية EditRecordLimited هامة في ذلك الوقت.

Publisher.vb

تتيح هذه الفورم للمستخدم تحرير سجلات جدول الناشرPublisher .وهي نموذج بسيط نسبياً مع حقلي إدخال للبيانات فقط يشير حقل "الحالة Status " إلى كيفية اتصال عدة سجلات"بند مسمى NamedItem " إلى هذا الناشر.يظهر زر صغير إلى اليسار من حقل إدخال النص من أجل موقع الناشر على الويب.وهذا الزر"أظهر موقع الويبShowWeb "،وعند نقره،فإنه يظهر صفحة الويب الموفرة إلى المستعرض الافتراضي للمستخدم.لتمكين هذا الزر،أضف الكود التالي على معالج حدث نقر زر"أظهر الويب".

```
' أظهر موقع الانترنت المعروض في الحقل
        Dim newProcess As ProcessStartInfo
        On Error Resume Next
        If (Trim(RecordWeb.Text) = "") Then Return
        newProcess = New ProcessStartInfo(Trim(RecordWeb.Text))
        Process.Start(newProcess)
```

SystemValue.vb

يعالج محرر هذا الكود البنود في جدولSystemValue .على الرغم من أننا سنصله إلى وصلة على نموذج المكتبة الرئيسي في هذا الفصل،سنعمل على تغيير طريقة إمكانية الوصول في فصل لاحق.

حسناً،هذه 11 فورم من 12 فورم مشتقة.الفورم الأخيرة هي الفورمNamedItem، المبينة في الشكل التالي.



إن الفورم NamedItemهي الفورم الأكبر والأكثر تعقيداً من بين النماذج المشتقة من النموذجBaseCodeForm. فهي تحرر بنود المكتبة الرئيسية المسجلة في جدول قاعدة البيانات NamedItem.وهي معقدة لأنها تدير بشكل مباشر أو غير مباشر السجلات في جداول تابعة (أو ثانوية)أخرى: ItemAuthor، ItemCopy، ItemKeyword، ItemSubject وبشكل غير مباشر الجداول: Author، Keyword، Publisher، و Subject.

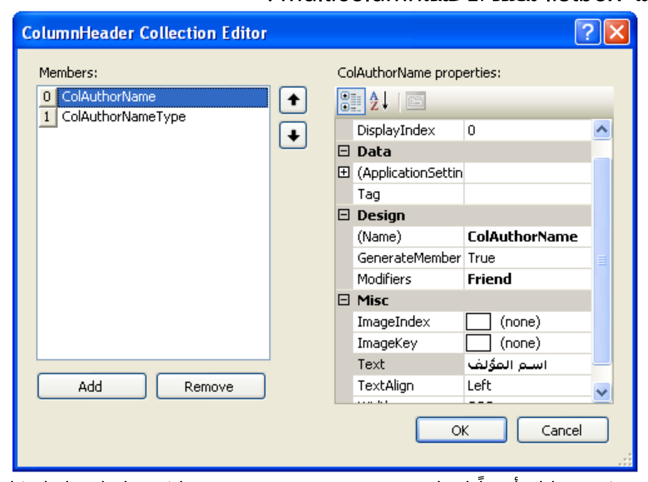
جميع الحقول على التبويب"عام" وتبويبات التصنيف تعتمد على حقول المدخلات التي تتدفق مباشرةً إلى جدولNamedItem، تماماً كما تم عمله مع نماذج تحرير السجلات الأخرى.تستخدم حقول السلاسل Seriesوالناشر Publisherنماذج اختيار منفصلة(PublisherAddLocateو SeriesAddLocate)للحصول على قيم المعرف IDالمخزنة في NamedItem.إليك الكود الذي يبحث عن الناشر:

```
' سؤال المستخدم
        newPublisher = (New PublisherAddLocate).PromptUser()
        If (newPublisher = -1) Then Return
        ' الاختيار لتنظيف الناشر
        If (newPublisher = -2) Then
            RecordPublisher.Text = "Not Available"
            PublisherID = -1
            Return
        End If
```

التبويبات الأربع الباقية- المؤلفين/الأسماءAuthors/Names ،المواضيعSubjects ،الكلمات المفتاحيةKeywords ،والنسخCopies -تدير السجلات الثانوية. الكود ثابت(أو متطابق) إلى حدما بين التبويبات الأربع،لذلك سوف سأقتصر في تعليقي على التبويب مؤلفين/أسماء(شاهد الشكل التالي).



الأدوات على هذا التبويب مشابهة لتلك التي على النموذج ListEditRecords،فهي موجودة لإدارة مجموعة من السجلات في جدول ما.في هذه الحالة،إنه الجدول ItemAuthor.من أجل تقديم القائمة،سأختار استخدام الأداة "عرض القائمة ListView" بدلاً من أداة صندوق القائمة ListBox القياسية.بوضع خاصية عرض View لأداة "عرض القائمة ListView " إلى "تفاصيلDetails" ،بوضع حقلها FullRowSelectإلى صوابTrue، وتعديل مجمع أعمدتها Columns (شاهد الشكل التالي)تستطيع بسرعة تبديلها إلى صندوق قائمة listbox متعدد الأعمدةmulticolumn .



عندما تعمل على إضافة بند ما إلى هذه القائمة،سيكون عليك أيضاً إضافة"بنود فرعيةsubitems" ليكون ليظهر لديك كل شيء ولكن جميعها في العمود الأول.

```
Dim newItem As Windows.Forms.ListViewItem = AuthorsList.Items.Add("John Smith")
newItem.SubItems.Add("Illustrator")
```

يعمل الزر "إضافة Add"على إظهار النموذجAuthorAddLocate، بينما يعرض الزر"خاصيات Properties " الفورم ItemAuthorEdit.

قبل أن يتم إضافة أي سجل ثانوي،يجب أن يتواجد السجل"الأبparent " في قاعدة البيانات.وهذا لأن السجلات"الأبناءchild " تتضمن رقم المعرف ID لسجل الأب،وبدون سجل الأب، فلايوجد رقم معرف IDالأب.

إذا نظرت على كل روتينات الزر "إضافة Add" على هذه الفورم،ستجد كود مشابه للتالي:

```
' يجب ان يتم حفظ السجل أولاً
        If (ActiveID = -1) Then
            ' التأكيد على المستخدم
            If (MsgBox("قبل أن يتم إضافة المؤلفين أو " & " يجب ان يتم حفظ البند إلى قاعدة البيانات
الأسماء " & "هل تريد حفظ السجل الآن؟", MsgBoxStyle.YesNo Or MsgBoxStyle.Question, _
            ProgramTitle) <> MsgBoxResult.Yes) Then Return
            ' التحقق وحفظ البيانات
            If (ValidateFormData() = False) Then Return
            If (SaveFormData() = False) Then Return
        End If
```

إذا كان هذا سجل NamedItemجديد تماماً(ActiveID = –1)فإن هذا الكود سيحفظه قبل السماح للمستخدم إضافة سجل تابع.أي بيانات غير صحيحة تمنع من أن يتم حفظ سجل سيتم التقاطها في الاستدعاء للروتين ValidateFormData الفعلي،والاستدعاءات لكلا الروتينين ValidateFormData وSaveFormDataيحدث نفس الشيء عندما ينقر المستخدم على الزر "موافق OK ".عادةً،يعمل هذا على إطلاق إعادة رقم معرف ID سجل جديد على النموذج المستدعي.ولكن ماذا بخصوص إذا ما تم استدعاء SaveFormDataبواسطة إضافة مؤلف،ولكن بعدها نقر المستخدم الزر"إلغاء Cancel"(والذي يعيد عادة القيمة 1- ليشير على"أنه لم يتم إضافة سجل")؟لتجنب ذلك،تعمل الدالة SaveFormDataعلى وضع متغير على مستوى الفئة مسمى SessionSaved.

```
SessionSaved = True
```

يتم تنظيف هذه العلامة عندما يتم فتح الفورم للمرة الأولى،ولكن يتم وضعها إلى صواب نسبياً في أي وقت يتغير سجل ثانوي.تعمل النموذج NamedItemعلى إعادة قيادة الدوال AddRecord وEditRecordوتتفحص هذه العلامة قبل إعادة قيمة إلى النموذج المستدعي.

```
If (Me.DialogResult = Windows.Forms.DialogResult.OK) Or (SessionSaved = True) Then Return
ActiveID Else Return -1
```

يوجد الكثير من الكود الهام في الفورمNamedItem .سأترك لك تفحص كودها.

Connecting the Editors to the Main Form.ربط المحررات إلى الفورم الرئيسي

إذا شغلت التطبيق الآن،لن ترى أي اختلاف على الإطلاق.ما نزال نحتاج إلى وصل جميع محررات السجلات إلى الفورم الرئيسي.جميعها ترتبط من خلال الأدوات LinkLabelعلى لوحة الإدارة للفورم الرئيسي(PanelAdmin).نحتاج إلى إضافة 12 معالج حدث نقر الوصلات LinkClicked للتمكن من الوصول إلى جميع النماذج المتنوعة الجديدة.تقدم للأمام وأضفها الآن إلى الفئة الرئيسية.

كل من معالجات حدث LinkClickedعلى الأغلب صورة عن الأخرى،ماعدا القليل من أسماء حالة كائن هنا وهناك.إليك الكود الذي يعالج النقر على وصلة الناشر:

```
Private Sub AdminLinkPublishers_LinkClicked(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) _
Handles AdminLinkPublishers.LinkClicked
                If (SecurityProfile(LibrarySecurity.ManagePublishers) = _
        False) Then
            MsgBox(NotAuthorizedMessage, MsgBoxStyle.OkOnly Or _
            MsgBoxStyle.Exclamation, ProgramTitle)
            Return
        End If
                ListEditRecords.ManageRecords(New Library.Publisher)
        ListEditRecords = Nothing
    End Sub
```

بعد عمل اختبار سريع للأمن،فإن الكود يستدعي نموذج ListEditRecords القياسي،ممرراً حالة من محرر سجلاته ليتم استخدامها.ماتزال هناك وصلات غير فعالة على لوحة الإدارة والتي سنعمل على تمكينها في فصول لاحقة.

Setting the Default Location.إعداد الموضع الافتراضي

إن البرنامج الآن جاهز للتشغيل مع جميع الميزات الجديدة.بما أننا عملنا على إضافة الميزات الإدارية،عليك نقر الزر "تسجيل الدخول Login " في الزاوية العلوية اليسارية للفورم الرئيسية قبل أن تتمكن من الوصول إلى لوحة الإدارة وميزاتها.مالم تغيره،فإن اسم مستخدم تسجيل الدخول هو "admin "بدون الحاجة إلى كتابة كلمة المرور.على الرغم من أنك تستطيع الآن تشغيل البرنامج والوصول إلى محررات السجلات،لن تكون قادر على إضافة نسخة بند جديد item copies حتى تضع الموضع الافتراضيdefault location .لوضع الموضع الافتراضي:

1.أضف على الأقل موضع واحد من خلال وصلة "المواضع Locations "على لوحة الإدارة.

2.احصل على رقم المعرف ID لسجل CodeLocationالذي تريد أن يكون افتراضي.

تستطيع استخدام ميزات الاستعلام لمنصة إدارة سكول سرفر السريعةSQL Server Management Studio Express للوصول إلى السجلات في هذا الجدول.إذا كانت هذه المرة الأولى التي تضيف فيها سجلات إلى جدولCodeLocation، فإن البند الأول سيكون له قيمة IDالرقم المعرف 1.

3.ارجع إلى برنامج المكتبة،حرر جدول SystemValueمن خلال وصلة قيم النظام System Values على لوحة الإدارة.

4.أضف أو عدل قيمة نظام" DefaultLocation "،وضع قيمتها إلى رقم المعرف ID لسجل الموضع الافتراضي .

بشكل بديل،تستطيع تحديث سجل"الموضع الافتراضي DefaultLocation "في جدول قيم النظام SystemValueمباشرةً باستخدام منصة إدارة سكول سرفر السريعةSQL Server Management Studio Express. إذا كان المعرف IDللموضع location المستخدم هو1،استخدم عبارة سكول هذه لعمل التغير:

```
Update(SystemValue)
SET ValueData = '1'
WHERE ValueName = 'DefaultLocation'
```

في فصل لاحق،سنعمل على إضافة طرق قريبة من المستخدم لتحديث هذا الموضع الافتراضي.

لغة الترميز القابلة للتوسع XML(Extensible Markup Language)

XMLهي محاولة لتنظيم البيانات في تركيب قابل للاستخدام من قبل الناس والبرمجيات،وقد راجت في السنين الأخيرة،ولكن جذورها قديمة جداً. فقد تم اشتقاقها من SGML(لغة الترميز المعممة المعيارية Standard Generalized Markup Language) وكذلك **HTML**(لغة ترميز النصوص الفائقة(المدمجة) (Hyper Text Markup Language) القريبة منها و SGMLبدورها أتت من GML(لغة الترميز المعممة Generalized Markup Language)،وهي "لغة التوصيف metalanguage (لغة تصف لغة أخرى) تم تصميمها بواسطة IBMفي عام1960.
تمثل أبجدية تقنيات معالجة البيانات،الأبجدية التي لديها سبع محاور *Xs* غريبة.

ماهي لغة الترميز القابلة للتوسع؟ What Is XML?

XMLهي لا شيء أكثر من تنسيق بيانات بحيث تكون قابلة للقراءة من قبل الإنسان وقابلة للقراء من قبل الآلة.هل جربت فتح مستند ميكروسوفت ورد بواسطة المفكرة؟شاهد الشكل التالي.على الرغم من أنك تستطيع عادةً التحقق sift outمن النص الرئيسي للمستند،فمعظم ما تراه غير مفهوم(هراء).وهذا لأنه في تنسيق ثنائي خاص proprietary binary format. وهو خاص proprietaryلأن،وبصراحة،ليس عليك حشر أصابعك فيه.وهذا ما عليه ميكروسوفت ورد.وهو ثنائي binaryلأنك تستطيع تخزين الكثير من المعلومات بشكل مناسب في مساحة صغيرة من القرص.بمثل هذا الملف،أستطيع أن أخزن بياناتي بأي طريقة أختارها.في الحقيقة أستطيع تدوين بياناتي طوعاً أو كرهاً willy-nilly، وليس عليّ الحصول على ترخيص من أي واحد،لأنها تخصني،جميعها لي.(يبين الشكل التالي محاولة فتح مستند ورد لهذا الفصل بواسطة المفكرة)

الملفات الثنائية Binary filesعظيمة من أجل تخزين أي نوع من البيانات:أعداد،سلاسل حرفية،صور مشفرة،جداول من الشبكات،أي شيء.المشكلة هي :إذا لم تعلم التركيب الدقيق الذي استخدمته لتدوينها،فهناك دائما فرصة قليلة في استعادة البيانات.وهذا جيد إذا كان هدفك السرية secrecy،ولكن إذا احتجت إلى مشاركة البيانات دائماً مع أشخاص أو برامج أخرى،أو حتى أسوء،تصحيح مخرجات من برنامج منحرف،فإنك في مأزق.فإذا ما تم فقد بايت صغير،فمن المحتمل أن يصبح كامل الملف غير مفيد.

بالطبع يوجد طرق أخرى لتخزين بياناتك.من أجل الملفات التي تخزن سجلات بيانات،توفر ملفات محدِّد الجدولة tab-delimited (محدد تنظيم الجدولة) و CSV(القيمة المفصولة بفاصلة comma-separated value) وسيلة نقل مريحة(مناسبة)،في تنسيق أكثر وداً للإنسان.على سبيل المثال،خذ هذه البيانات من عينة قاعدة بيانات ميكروسوفت التجارية"نورثويند Northwind "،فقد تم تخزينها كقيم مفصولة بفاصلة comma-separated values .

```
ProductID,ProductName,SupplierID,Category,UnitPrice,Available
"1"،"Chai","652","Beverages","$18.00","Yes"
"2"،"Chang","9874","Beverages","$19.00","No"
"3"،"Aniseed Syrup","9874","Condiments","On Sale","Yes"
```

الآن هذا أفضل. هذه البيانات سهلة الفهم نوعاً ما.كل قطعة من البيانات تم تجميعها بواسطة فواصل،والصف الأول يشير إلى ما يحتويه كل عمود.والجزء الأفضل أن العديد من البرامج تعرف سابقاً كيف تقرأ ملفات في هذا التنسيق.إذا حفظت هذه البيانات في ملف نصي بالامتداد *.csv*، وفتحتها باستخدام ميكروسوفت اكسل،فإن البيانات ستظهر بشكل آلي في أعمدة.

ولكن يمكن أن تكون أفضل.على سبيل المثال،إلى ما تشير هذه القيم"652" و"9874" وبأي طريقة؟وهل صحيح أن سعر الوحدة لـ Aniseed Syrup هي "برسم البيع On Sale" ؟

بالتأكيد أستطيع تحميل هذه البيانات في برنامجي،ولكن هل استطيع فعل أي شيء بها؟على الأقل هي سهلة القراءة لكل من الإنسان وبرامج الكمبيوتر،و أليس هذا ما قلته حول XML ؟

حسناً، على الرغم من أن XMLتتضمن قواعد وميزات تجعلها أكثر مرونة من ملف بيانات نصي عادي،وهذا ليس هو الفرق.من أجل جميع المبالغات،فإن XMLهي مجرد طريقة في تخزين البيانات.أي سمة لـ XMLتم مناقشتها في هذا الفصل يمكن أن يتم إنجازها بسهولة بواسطة البيانات المخزنة في نص أبسط simpler text أو تنسيقات خاصة ثنائية binary proprietary formats. في الحقيقة،على الأغلب من الأسرع وأكثر راحة للتطوير استخدام تنسيق خاص proprietary،لأن بياناتك ستحتوي على ما تحتاجه فقط وبدقة ،دون أي خطأ.

يمكن القول أن XML تتضمن عدة سمات والتي تجعل منها منافس قوي contender عند اعتبار تنسيق البيانات:

إنها بسيطة بالنسبة للقراءة *It's straightforward to read*

يتضمن كل عنصر نوع بيانات عنوان،والعناوين الجيدة تجعل القراءة جيدة.

سهلة المعالجة *It's easy to process*

جميع البيانات تتضمن وسم بداية ونهاية،لذلك يمكن للبرنامج أن يعالج البيانات دون الكثير من الجهد.وعنصر واحد سيء لن يعمل بالضرورة على إتلاف ruin كامل الملف.

مرنة *It's flexible*

تستطيع تخزين أي نوع من البيانات في XML. بالنتيجة فهو مجرد ملف نصي.إذا كان لديك تنسيق ملف XMLمعين مستخدم في النسخة رقم1 من برنامجك،وأضفت ميزات له في النسخة رقم2،تستطيع عمله بطريقة بحيث يبقى متاح لبرامج النسخة1 استخدام ملفات النسخة2 بدون خرق.

تصف نفسها *It's self-describing*

تتضمن XMLالعديد من الميزات والتي تتيح لك وصف محتوى ملفXML . الأكثر شيوعاً اثنان هما DTD (تعريف نوع المستند Document Type Definition) وXSD (تعريف تصميم بياناتXML (XML Schema Definition)).إنك تستخدم هذه الأدوات للإشارة إلى ما تتوقع أن يحويه بالضبط ملف بياناتك.بالإضافة لذلك،تسمح XMLلك تضمين تعليقات في المحتوى دون التأثير على البيانات الفعلية.

ذاتية التحقق *It's self-verifying*

الأدوات المتاحة من ضمنها الأدوات في الدوت نت،والتي تستطيع التشديد على تكامل وتنسيق ملف XMLبمقارنة المحتوى لـ DTDو XSD المصاحب .وهذا يتيح لك التحقق من ملف قبل حتى معالجته.

إنها معيار مفتوح *It's an open standard*

لقد حصلت XMLعلى قبول واسع الانتشارwidespread acceptance ،حتى عبر منصات الكمبيوتر المتشعبةdivergent computer platforms .

إنها مبنية(جاهزة) في الدوت نت *It's built into .NET*

هذا سيكون السبب الأعظم لاستخدامها.في الحقيقة،لن تكون قادر على الابتعاد عن XMLفي الدوت نت،حتى لو حاولت.فهي في كل مكان.

ولكن توجد أخبار سيئة أيضاً:

إنها ضخمة(كبيرة الحجم) *It's bulky*

يحتوي محتوى XMLعلى الكثير من المعلومات التركيبية المتكررة،وبشكل عام الكثير من الفراغات البيضاء.تستطيع اختصار abbreviateالعديد من عناصر التركيب،وإزالة جميع المساحة البيضاءwhitespace (لا تحتاجها XML)،ولكن هذا سيزيل سمات قابلية القراءة بالنسبة للإنسان من البيانات.بعض منصات العملplatforms ، مثل مستعرضات الهواتف الخلويةcell phone browser،تحب الاحتفاظ بالبيانات صغيرة. XMLهي أي شيء ولكن صغيرة.

إنها نص *It's text*

انتظر هذا شيء جيد في معظم الأوقات.بعض الأحيان تريد تخزين بيانات ثنائيةbinary فقط،مثل الصور.في الواقع لاتستطيع تخزين بيانات ثنائية حقيقية في ملف XMLدون كسر واحد من القواعد الأساسية لـ XML تقريباً:نصوص فقط! على الأغلب،البيانات الثنائية يتم تشفيرها في تنسيق يشبه النص،مثل القاعدة-64(والتي تستخدم الحروف القابلة للقراءة لتخزين البيانات الثنائية)

إنها غير فعالة *It's inefficient*

وهذا يأتي من كون لديها بيانات في تنسيق شبه قابل للقراءة من قبل الإنسان كثير الحشو(مسهب)،بدلاً من المصقول(الوجيز)،وضغط النموذج الثنائي.ببساطة تستغرق وقت أطول بالنسبة للكمبيوتر لعمل مسح نص في البحث عن الأقواس المحدبة المتطابقة من عمل نقل عدة بايتات مباشرةً من كتلة بيانات ثنائية ضمن موقع في الذاكرة.

إنها قابلة للقراءة من قبل الإنسان *It's human-readable*

لايوجد الكثير من الأسرار في ملف XML.على الرغم من أنك تستطيع تشفير عناصر البيانات في الملف،أو كامل الملف لسبب ما،هذا سيؤدي نوعاً ما إلى إحباط الهدف من استخدام XML.

إنها قابلة للقراءة من قبل الآلة.*It's machine-readable*

XMLليست مناسبة لكل نوع ملف بيانات.

ليست معفية(منيعة) من الأخطاء. *It's not immune to errors*

كما أكرر،إن XML مجرد ملف نصي.فهي ليست الدواء العام،مجرد تنسيق ملف مفيد.

مبدأ لغة الترميز القابلة للتوسع.The XML Rule

قبل أن نلقي نظرة على XML الحقيقي،تحتاج إلى معرفة المبدأ.عليك أن تطيع obeyالقاعدة في كل قطعة من نص XMLالذي تكتبه.

القاعدة THE RULE

إذا فتحتها،أغلقها. If you open it, close it.

هذه هي لاتنسها.أطعها.وسأشرح ما تعني فيما بعد.

محتوى XML .XML Content

إذا لم تستخدم XML،ولكن قد كتبت بعض HTML،فإن ما أكتبه سيكون معروف نوعاً ما.

بعض قواعد Some Basic XML

إليك قطعة من XML لتستمتع بها.

```
<?xml version="1.0"?>
<hello>
    <there>
        <!-- Finally, real data here. -->
        <world target="everyone">think XML</world>
        <totalCount>694.34</totalCount>
        <goodbye />
```

```
    </there>
</hello>
```

إنها بيانات وبيانات مفيدة وإليك الأسباب:

إنها XML بوضوح. It's obviously XML

وهذا واضح من السطر الأول،والذي دائماً يبدأ بـ...xml?> ، هذا السطر يشير أيضاً إلى رقم نسخة XML،والتي تخبر روتين معالجة XML(المعربات parsers)لضبط السلوك عند الحاجة.

إنها هيكلية It's structured

إن XMLهو تركيب بيانات هرمي.تستطيع الحصول على عناصر البيانات المضمنة داخل عناصر بيانات أخرى لأي عمق تريد.كل عنصر محصور بمجموعة من الوسوم،في هذه العينة،الوسوم هي hello، there، world، totalCount، و goodbye.تظهر الوسوم دائماً ضمن<أقواس محدبةangle brackets >، ودائماً تظهر بشكل زوجي،كما في<hello>...</hello>. (ومن هنا أتت القاعدة"إذا فتحته،أغلقه")لاتنس " / " قبل اسم الوسم في قوس الإغلاق.
هذا البناء يتح لك تنظيم بياناتك في وحدات مسماة منظمة بشكل خاص.من أجل الوسوم التي ليس لديها شيء بين البداية والنهاية،تستطيع استخدام البناء المختزل(المختصرshortened syntax) <*tagname* />،كما فعلت مع الوسم goodbye.
على فكرةBy the way،وسوم XMLحساسة لحالة الأحرفcase-sensitive، لذلك اكتب بحذر.

إنها قابلة للقراءة It's readable

إنها قابلة للقراءة من قبل الإنسان،بفضل جميع الفراغات البيضاء،على الرغم من أنك تستطيع إزالة هذه الفراغات وسيبقى لديك XML. وهي أيضاً قابلة للقراءة من قبل الكمبيوتر بسبب الاستخدام الراسخ consistentللوسوم.

إنها وحدة مفردة من البيانات It's a single unit of data

جميع ملفات XMLلديها عنصر جذري *root element*مفرد والذي يجب أن تظهر فيه جميع العناصر الأخرى.في العينة،<hello> هو العنصر الجذري(الإقلاعي).حالما يتم إغلاق ذلك العنصر(من خلال وسم النهاية له)لاتستطيع إضافة أي عناصر إضافية.

إنها تحوي التعليقات It's got comments

شاهد السطر `<!-- Finally, real data here. -->`،إنه تعليق.تستطيع لصق التعليقات هنا وكأنها وسوم حرة الحركة free-floating tags.

إنها تحوي(تحصل على)المواصفات.It's got attributes

تدعم XMLتنوعين varietiesمن البيانات:real data البيانات الحقيقية والمواصفات.attributes.قيم البيانات الحقيقية تأتي بين صلب innermost أزواج الوسوم،كما مع`think XML` و `694.34` في العينة.توفر المواصفات معلومات موسعة حول الوسوم نفسها.لقد ضمنت مواصفة attributeمسماة `target` في العنصر world.إن المحتوى لجميع المواصفات يجب أن يكون ضمن علامتي اقتباس("").كان باستطاعتي جعل هذه المواصفة عنصر فرعي subelement بدلاً عن ذلك،والكثير من الناس يعملون ذلك. يوجد عدم اتفاق disagreementبين المبرمجين بخصوص متى يجب أن تكون البيانات عنصر أو مواصفة دع ضميرك conscience (قلبك) يقودك(يدلك).

بعض القواعد- والأهمية لـ XML. Some Basic—and Meaningful—XML

دعنا نرى كيف تبدو البيانات المفصولة بفاصلة من قاعدة بيانات "نورثويند " Northwind التجارية التي ذكرناها سابقاً في XML.

```
<?xml version="1.0"?>
<productList>
    <supplier ID="652" fullName="Beverages R Us">
        <product ID="1" available="Yes">
            <productName>Chai</productName>
            <category>Beverages</category>
            <unitPrice>18.00</unitPrice>
        </product>
    </supplier>
    <supplier ID="9874" fullName="We Sell Food">
        <product ID="2" available="No">
            <productName>Chang</productName>
            <category>Beverages</category>
            <unitPrice>19.00</unitPrice>
        </product>
        <product ID="3" available="Yes" onSale="true">
            <productName>Aniseed Syrup</productName>
            <category>Condiments</category>
            <unitPrice>12.00</unitPrice>
        </product>
    </supplier>
</productList>
```

نقل البيانات إلى XMLقد زاد بشكل كبير حجم المحتوى.ولكن مع الزيادة في الحجم تأتي الزيادة في القيمة المعالجة.لقد كان باستطاعتي بشكل مباشر الحصول على بعض الفائدة من تركيب XML الهيكلي.في البيانات الأصلية،المزود supplier كان عمود أخر فقط،ولكن في نسخة XML،جميع البيانات تم تجميعها الآن ضمن مقاطع المزودsupplier، والتي تكون مفهومة(على الأقل،إذا ما كانت هذه هي الكيفية التي عملت على تخطيطها لاستخدام البيانات).
ويمكنك أن ترى أني قد اتبعت القاعدة،كل سمة مفتوحة لديها سمة إغلاق مطابقة.مهما تعمل لاتنس القاعدة.

ماذا حول القسم القابل للقراءة من قبل الإنسان؟What About the Human-Readable Part?

واحدة من الأدوات المستخدمة مع XMLهي*XSLT*، والتي تحل محل Transformations تحويلات XSL (XSLتحل محل لغة الصفيحة النمطية القابلة للتوسعeXtensible Stylesheet Language).إن من الصعب على XSLT استخدام لغة الصياغة scripting language والتي تتيح لك تحويل بعض بيانات XMLإلى أي نوع بيانات آخر،أو تنسيق مخرجات تريده أنت.فهي مجرد مقدار قليل من اللغات المتعلقة بـ XSL المنشئة لمعالجة بيانات XMLبطرق معقدة.لنرى عمل XSL،خذ القطعة المفيدة من XMLالمجدولة سابقاً(العينة <productList>)وبدل السطر الأول ?xml بالأسطر التالية:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="hello.xsl"?>
```

احفظ نص إلى ملف على سطح مكتبك كـ hello.xml مايليه:ضع صيغة XSLTالتالية في ملف آخر على سطح مكتبك سمه hello.xsl.

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
    <xsl:template match="/">
        <xsl:text>
            ProductID,ProductName,SupplierID,Category,UnitPrice,Available
        </xsl:text>
        <BR/>
        <xsl:apply-templates/>
    </xsl:template>
    <xsl:template match="supplier">
        <xsl:variable name="supID" select="@ID"/>
        <xsl:for-each select="product">"
            <xsl:value-of select="@ID"/> ","
            <xsl:value-of select="productName"/> ",
            <xsl:value-of select="$supID"/> ","
            <xsl:value-of select="category"/> ","
            <xsl:choose>
                <xsl:when test="@onSale='true'">On Sale</xsl:when>
                <xsl:otherwise>$
                    <xsl:value-of select="unitPrice"/>
                </xsl:otherwise>
            </xsl:choose> ","
            <xsl:value-of select="@available"/> "
            <BR/>
        </xsl:for-each>
    </xsl:template>
</xsl:stylesheet>
```

لقد أخبرتك أنه صعب الاستخدام،وحتى الأصعب النظر إليه.والآن من أجل العرض.لدي مستكشف انترنت Internet Explorer منصب على نظامي،ولكن هذا سيعمل مع أغلب المستعرضات الحديثة.افتح ملف hello.xml في مستعرضك،سيظهر النص المنسق الجميل التالي:

```
ProductID,ProductName,SupplierID,Category, UnitPrice,Available
"1", "Chai", "652", "Beverages", " $18.00", "Yes"
"2", "Chang", "9874", "Beverages", " $19.00", "No"
"3", "Aniseed Syrup", "9874", "Condiments", "On Sale", "Yes"
```

والآن ،هذا محبب أكثر،حيث اجتمعت كل من XML وXSLTمع بعضهما وصنعا هذا التقدم في تقنية البيانات الممكنة.(لقد عملت خدعة صغيرة في هذا المثال،ستلاحظ المدخلات
 في صيغة XSLTوالتي لاتظهر في المخرجات النهائية.وقد عملت هذا لتبدو مناسبة في مستعرضك)،ولكن بشكل جدي،على الرغم من أنني كنت قادر على توليد مجموعة بيانات مفصولة بفاصلة comma-separated data set من خلال XSLT، فإن مهمات أكثر شيوعاً لـ XSLT تتضمن توليد HTMLمنسقة بشكل جميل بالاعتماد على بيانات XML،أو توليد مستند XML ذو عرض بديل خاص للبيانات الأصلية.كيف يعمل؟
بشكل أساسي، تخبر العناصر <xsl:template> المفسر parserللبحث عن وسوم في مستند XML والذي يطابق نموذج(عينة) pattern ما (مثل supplier).عندما يجد مطابقة،فإنه يقدم كل شيء داخل الوسوم <xsl:template>إلى وسم XMLالمطابق ومحتوياته.العينة(النموذج)المخصص في المواصفات match تستخدم تقنية XMLتدعى*XPath*: نظام للبحث بشكل عام عن الوسوم المتطابقة ضمن مستند XML.
من الظاهر أنه مربك!هو كذلك.

مخططات XML. XML Schemas

تتيح لك *XSD* (تعاريف تركيب XML، XML Structure Definitions) تعريف مخطط إما "لغة"language " أو "مفردات"vocabulary " لمستند XMLخاص بك.تذكر،أن XMLمعيار شامل مفتوح وواسع،تستطيع تعريف الوسوم بأي طريقة تريد ولن يأبه أحد،على الأقل حتى يصبح من الواجب عليك معالجة الوسوم ضمن برمجياتك.فإذا لم تكن صحيحة،فمن المحتمل أن تفشل معالجتك.تتيح لك XSDتعريف القواعد التي يجب أن يتبعها مستندك XMLإذا كان يجب اعتباره مستند صحيح من أجل أهدافك.
DTD) تعريف نوع مستندDocument Type Definition، مشابه لـXSD ، مع أنه تقنية أقدم.وهو مدعوم بشكل واسع من قبل أدوات XML،ولكنه ليس بتلك المرونة التي عليهاXSD. يوجد أيضاً لغات تعريف مخطط أخرى مشابهة لـ XSD،ولكن بما أن XSDمبني داخل الدوت نت فإننا سنركز عليه.)
مخططاتXSD هي كل بت مقدر(مثمنendearing)كما هو الحال في صيغ XSLT.لنعمل على إنشاء XSDمن أجل مثالنا XML <productList> الذي ذكرناه فيما سبق.أولاً،نحتاج إلى تغيير أعلى XMLلنسمح له معرفة أن ملف XSDهو المتاح.غير التالي:

```
<?"xml version="1.0?>
<productList>
```

إلى:

```
<?"xml version="1.0?>
```

```
"productList xmlns="SimpleProductList>
"xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
<"xsi:noNamespaceSchemaLocation="hello.xsd
```

هذه التعليمات(التوجيهات directives)تخبر مفسر XMLللبحث في مخطط hello.xsd .وهي تعرف أيضاً فضاء أسماء،فيما بعد.يحتوي ملف hello.xsd المخطط التالي:

```
"xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema>
<"targetNamespace="SimpleProductList
</"xs:element name="productList" type="ProductListType>
<"xs:complexType name="ProductListType>
<xs:sequence>
"xs:element name="supplier" type="SupplierType>
</"maxOccurs="unbounded
<xs:sequence/>
<xs:complexType/>
<"xs:complexType name="SupplierType>
<xs:sequence>
"xs:element name="product" type="ProductType>
</"maxOccurs="unbounded
<xs:sequence/>
</"xs:attribute name="ID" type="xs:integer>
</"xs:attribute name="fullName" type="xs:string>
<xs:complexType/>
<"xs:complexType name="ProductType>
<xs:sequence>
</"xs:element name="productName" type="xs:string>
</"xs:element name="category" type="xs:string>
</"xs:element name="unitPrice" type="xs:decimal>
<xs:sequence/>
</"xs:attribute name="ID" type="xs:integer>
</"xs:attribute name="available" type="YesOrNoType>
</"xs:attribute name="onSale" type="xs:boolean>
<xs:complexType/>
<"xs:simpleType name="YesOrNoType>
<"xs:restriction base="xs:string>
</"xs:enumeration value="Yes>
</"xs:enumeration value="No>
<xs:restriction/>
<xs:simpleType/>
<xs:schema/>
```

يبدو أنه متداخل،أليس كذلك؟عملياً إنه أكثر بساطة من XSLT.بشكل أساسي،يقول المخطط أنه من أجل كل عنصر(أو "وسمtag " أو "عقدةnode ")في مستنديXML ،هنا توجد العناصر الفرعية subelementsوالمواصفات التي تحويهاattributes ،ونوع البيانات لكل منهم.تستطيع حتى إنشاء أنواع بيانات مستعارة(أو زائفةpseudodata) خاصة بك(عملياً،العوامل المقيدة limiting factors على أنواع البيانات المتواجدة)،كما فعلت مع نوع البيانات YesOrNoType،والذي يقيد القيمة المتعلقة إلى نصوص نعم Yesو لا No.

تستطيع النظر إلى ملف XMLمع مخطط XSDالملحق(المرافق) في مستعرضك.ولكنه لن يكون بكل تلك الأهمية.إنه يريك فقط الـ XML.ولكن المخططات ستكون مفيدة عندما تحتاج تقييم(تعيين أهمية) النوعية بالنسبة لبيانات XMLالتي تأتي ضمن تطبيقات برمجياتك من مصادر خارجية.

فضاءات أسماء XML. XML Namespaces

يمكن أن تختلف قائمة المنتج product list لـ XMLالتي عرضتها سابقاً،فقد تكون ذات اسم مختلف وقواعد تنسيق مختلفة أيضاً،على سبيل المثال،يمكن أن يتم إنشاء مستند بحيث يبدو كالتالي:

```
<?"xml version="1.0?>
<allProducts>
<"vendor ID="652" vendorName="Beverages R Us>
<"item ID="1" available="Yes>
<itemName>Chai</itemName>
<group>Beverages</group>
<priceEach>18.00</priceEach>
<item/>
```

```
</vendor>
</allProducts>
```

جميع البيانات نفسها،ولكن الوسوم tagsمختلفة.مستند مثل هذا سيكون متضارب incompatibleمع برمجيات مكتوبة للعمل مع مستندنا الأصلي.فتشغيل المستند من خلال XSDالخاص بنا سوف يخبرنا بسرعة أن لدينا أخطاء في مجموعة البيانات bogus data set،ولكنه سيكون من الجميل لو أخبرنا شيء ما هذا من البداية.أدخل فضاءات الأسماء*namespaces* .

توفر فضاءات الأسماء *namespaces* طريقة مريحة لقول"هذا الوسم الخصوصي في مستند XML يستخدم لغة التعريف XSD-defined language هذه."لاحظ بداية مخطط XSDالمبين سابقاً:

```
"xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema>
```

هذا السطر يثبت فضاء الأسماء المسمى xsباستخدام المواصفة xmlns.(الجزء xs:يخبر XMLماذا تريد أن تسمي فضاء أسماءك.)قيمة المواصفة هي معرف(محدد) مورد موحد(Uniform Resource Identifier (URI ، مجرد قيمة مفردة ومميزة بحيث تكون واثق من أنه لن يستخدمها شخص أخر.

نموذجياً،تستخدم عنوان موقع ويب من أجل شركتك،ليس من الواجب أن يكون الموقع موجود.تستطيع حتى وضع رقم هاتفك هناك،مجرد كونها قيمة مفردة ومميزة.

الطريقة الأكثر شيوعاً في استخدام فضاء أسماء هي سبق الوسوم ذات الصلة في مستند XML باسم فضاء الأسماء الجديد،كما في" xs:schema "عوضاً عن " schema "فقط.وهذا يقول للمفسر"إذا كنت تختبر بناءي على مخطط XSD،استخدم الاسم الذي حددته لفضاء الأسماءxs ." تستطيع أيضاً استخدام فضاء الأسماء "الافتراضيdefault " أيضاً من أجل عنصر معطى وجميع ما ينحدر منه descendantsبتضمين المواصفة xmlnsفي عنصر خارجي.ومن ثم جميع العناصر ضمن ذلك العنصر الخارجي سيستخدم فضاء الأسماء المخصص.ولقد استخدمت هذه الطريقة في واحد من الأمثلة السابقة:

```
<productList xmlns="SimpleProductList"...
```

من أجل ملفات XMLالقاعدية والتي سيتم استخدامها فقط بواسطة برنامجك،من المحتمل أن لاتحتاج التعب بفضاءات الأسماء.في الحقيقة تأتي في المتناول عندما تعمل على إنشاء بيانات XML والتي تستخدم معيار توزيع عمومي publicly published standard نوعاً ما.يوجد أيضاً حالات (نسخ) بحيث ومن المحتمل أن يحتوي ملف XMLمفرد بيانات ذات صلة باستخدامات متمايزة لاثنان أو أكثر من XML.في هذه الحالة،الأجزاء المختلفة من ملف XMLيمكن أن تشير إلى فضاءات أسماء مختلفةnamespaces .

بالنسبة للأجزاء الأخرى من عالم XML ، XSDوفضاءات الأسماء ليست سهلة الاستخدام،ولكنها مرنة flexibleوقويةpowerful .وكالعادة،توجد أدوات tools،أدوات مضمنة في الفيجوال أستوديو،والتي تتيح لك بناءها جميعاً دون أن يكون عليك التفكير بخصوص التفاصيل.

كما أقول دائماً،إن XMLهي مجرد بيانات، XMLوتقنياتها ذات الصلة توفر طريقة للمساعدة في ضمان ensureأن بياناتك جاهزة للاستخدام في تطبيقك.

استخدام XMLفي الدوت نت:الطريقة القديمةUsing XML in .NET: The Old Way

تتضمن الفيجوال بيسك طريقتين للعمل مع محتوى XML: الطريقة القديمة والطريقة الجديدة.تستخدم الطريقة القديمة فئات من فضاء الأسماءSystem.Xml ، وتوفر وصول تقليدي معتمد على كائن object-basedإلى وسوم XML،المواصفاتattributes ،والبياناتdata. الطريقة الجديدة،تم تقديمها في الإصدار2008،تستخدم الفئات في فضاء الأسماء System.Xml.Linqوتوفر وصول إلى محتوى XMLمباشرةً ضمن بناء الكود المصدري للفيجوال بيسك.سأناقش كل من الطريقتين في هذا الفصل.

بما أن جودة إدارة XMLليست بقدر ضخامة مقطع نصه،تتضمن الدوت نت العديد من الفئات التي تدير بيانات XML.جميع هذه الأدوات تظهر في فضاء الأسماء System.Xml وفضاءات الأسماء التابعة له:

System.Xml

التجميع الرئيسي للفئات ذات الصلة بـ XML في الطريقة القديمةold-way XML-related classes .

System.Xml.Linq

الفئات التي تكامل(تدمجintegrate) XMLمع تقنيات لينكوLINQ. وهذه هي الطريقة الجديدة والتي سأتحدث عنها لاحقاً.

System.Xml.Schema

الفئات التي تنشئ وتستخدم مخططات XSD.

System.Xml.Serialization

الفئات التي تقرأ وتكتب مستندات XMLبواسطة جدولة دوت نت القياسيةstandard .NET stream .

System.Xml.XPath

الفئات التي تنفذ تقنية XPathالمستخدمة لبحث مستندات XML.

System.Xml.Xsl

الفئات التي تمكن تحويلات XSL.

الميزات المضمنة في كل فئة ترتبط بإحكام إلى تركيب XMLوالتقنيات ذات الصلة مثل XSLT وXSD .

فئات XMLالقاعدية،قواعدياً.The Basic XML Classes, Basically

يتضمن فضاء الأسماء The System.Xml أغلب الفئات القاعدية التي ستستخدمها لإدارة بيانات XML. إن الكائن XmlDocumentهو العرض في الذاكرة لمستندك XMLالفعلي:

```
Dim myData As New System.Xml.XmlDocument
```

إن مستندك مركب من التصريحات declarations (ذلك الشيء<?xml...?> في الأعلى)،عناصر البياناتdata elements (جميع الوسوم tagsالنوعية في مستندك)،المواصفات attributes (داخل كل بداية وسم عنصر)،والتعليقاتcomments. وهذا كله يتم تمثيله بواسطة الفئات XmlDeclaration، XmlElement،XmlAttribute ، XmlCommentعلى الترتيب.مع بعضها هذه الوحدات الأربع لمستندك تدعى "العقد *nodes* "،ممثلة بشكل عام بواسطة الفئة .(جميع الفئات الأربع ترث من الفئة القاعدية XmlNode الأكبر).عندما تبني مستند XMLيدوياً في الذاكرة،فإنك تستخدم فئات مستقلة مثل XmlElement.بعدئذٍ، عندما تحتاج لعمل مسح خلال مستند موجود،من الأسهل استخدام الفئة XmlNodeالشاملة.

دعنا نبني مجموعة جزئية من مثال بيانات المنتج product data لـXML .

```
<?xml version="1.0"?>
<productList>
<!-- We currently sell these items. -->
```

```
<supplier ID="652" fullName="Beverages R Us">
<product ID="1" available="Yes">
<productName>Chai</productName>
<category>Beverages</category>
<unitPrice>18.00</unitPrice>
</product>
</supplier>
</productList>
```

صرح عن جميع المتغيرات التي ستستخدمها،ومن ثم استخدمها.

```
Dim products As Xml.XmlDocument
    Dim prodDeclare As Xml.XmlDeclaration
    Dim rootSet As Xml.XmlElement
    Dim supplier As Xml.XmlElement
    Dim product As Xml.XmlElement
    Dim productValue As Xml.XmlElement
    Dim comment As Xml.XmlComment

        ' ----- Create the document with a valid declaration.
        products = New Xml.XmlDocument
        prodDeclare = products.CreateXmlDeclaration("1.0", _
        Nothing, String.Empty)
        products.InsertBefore(prodDeclare, products.DocumentElement)
        ' ----- Create the root element, <productList>.
        rootSet = products.CreateElement("productList")
        products.InsertAfter(rootSet, prodDeclare)
        ' ----- Add a nice comment.
        comment = products.CreateComment( _
        " We currently sell these items. ")
        rootSet.AppendChild(comment)
        ' ------ Create the supplier element, <supplier>.
        ' Include the attributes.
        supplier = products.CreateElement("supplier")
        supplier.SetAttribute("ID", "652")
        supplier.SetAttribute("fullName", "Beverages R Us")
        rootSet.AppendChild(supplier)
        ' ----- Create the product element, <product>, with the
        ' subordinate data values.
        product = products.CreateElement("product")
        product.SetAttribute("ID", "1")
        product.SetAttribute("available", "yes")
        supplier.AppendChild(product)
        productValue = products.CreateElement("productName")
        productValue.InnerText = "Chai"
        product.AppendChild(productValue)
        productValue = products.CreateElement("category")
        productValue.InnerText = "Beverages"
        product.AppendChild(productValue)
        productValue = products.CreateElement("unitPrice")
        productValue.InnerText = "18.00"
        product.AppendChild(productValue)
```

إنه يعمل حقاً،لبرهان ذلك،ضع هذا الكود في حدث النقر لزر ما،وأنهيه بالسطر التالي(أي أضف السطر التالي لنهاية الكود السابق).

```
products.Save("c:\products.xml")
```

شغل البرنامج واعرض الملف *c:\products.xml*لرؤية XML product data.توجد عدة طرق مختلفة لاستخدام فئات XML لإنشاء مستند XMLفي الذاكرة.على سبيل المثال،على الرغم من أنني استخدمت الطريقة SetAttributeلإضافة مواصفات إلى عقدة المزود supplierوالمنتجproduct ،فقد أستطيع إنشاء كائنات مواصفات منفصلة،وتزيلهم على هذه العقد،تماما مثلما فعلت من أجل العنصر الرئيسي.

```
Dim attrData As XmlAttribute
        attrData = products.CreateAttribute("ID")
        attrData.Value = "652"
        supplier.SetAttributeNode(attrData)
```

إذاً،هذا جميل ،ولكن ماذا بخصوص إذا ما كان لديك بعض XMLفي ملف ما،وتريد تحميلها في كائن XmlDocument؟ببساطة استخدم الطريقة Load للكائن XmlDocument.

```
Dim products As XmlDocument
        products = New XmlDocument
        products.Load("c:\products.xml")
```

من أجل تلك النسخ(الحالات)التي تريد فقط القراءة أو كتابة بعض XMLمن أو إلى ملف ما،ولا تأبه كثيراً بخصوص معالجتها في الذاكرة،تتيح لك الفئة XmlTextReaderوالفئة XmlTextWriter قراءة وكتابة بيانات XMLبسرعة بواسطة جدولة النص text stream. ولكن إذا كنت ذاهب لعمل أشياء ببيانات XML في برنامجك،فإن الطرق Loadو Save لكائن XmlDocumentهي الخيار الأفضل.

إيجاد الإبر وكومات القش Finding Needles and Haystacks

في مثال البيانات خاصتنا،جميع المنتجات productsتظهر في مجموعات المزود.إذا كنا نريد فقط قائمة بالمنتجات products،بغض النظر عن المزود supplier،نطلب من الكائن XmlDocumentتزويد تلك البيانات بواسطة الكائن XmlNodeList.

```
Dim justProducts As XmlNodeList
        Dim oneProduct As XmlNode
        ' ----- First, get the list.
        justProducts = products.GetElementsByTagName("product")
        ' ----- Then do something with them.
        For Each oneProduct In justProducts
            ' ----- Put interesting code here.
        Next oneProduct
        MsgBox("Processed " & justProducts.Count.ToString() & " product(s).")
```

من أجل اختيار أكثر تعقيداً للعقد ضمن المستند،ينفذ فضاء الأسماء System.Xml.XPath لغة بحثXPath ، والتي تمنحك مرونة زائدة في البنود المستكشفة. تصف مجموعة مستندات الفيجوال أستوديو بناء الطرق والبحث المستخدم مع هذه الفئات.

مخطط التحقق(الإثبات).Schema Verification

يستطيع الكائن XmlDocument الاحتفاظ بأي نوع من محتوى XML المحقق والعشوائي،ولكن تستطيع أيضاً التحقق من المستند على مخطط XSD.إذا كان مستندك XML يشير إلى مخطط XSD،أو يتضمن DTD،أو يستخدم XDR(مخططات بيانات XML المختزلة XML Data Reduced schemas،مشابهة لـ XSD)،أو XmlReader،عند تركيبه بواسطة إعدادات قارئ XmlReaderSettings مناسب،سيقارن بدقة بيانات XML على القواعد المعرفة،ويرمي باستثناء إذا كانت هناك مشكلة.

```
Dim products As New XmlDocument
        Dim xmlRead As XmlTextReader
        Dim withVerify As New XmlReaderSettings
        Dim xmlReadGood As XmlReader
        ' ----- Open the XML file and process schemas
        ' referenced within the content.
        withVerify.ValidationType = ValidationType.Schema
        xmlRead = New XmlTextReader("c:\temp\products.xml")
        xmlReadGood = XmlReader.Create(xmlRead, withVerify)
        ' ----- Load content, or throw exception on
        ' validation failure.
        products.Load(xmlReadGood)
        ' ----- Clean up.
        xmlReadGood.Close()
        xmlRead.Close()
```

تحويلات XML. XML Transformations

تحويلات XSL ليست أكثر صعوبة من أية معالجات أخرى لـ XML. كما يوجد العديد من الطرق للحصول على بيانات مصدرية لـ XML(من ملف،بناءه بشكل يدوي بواسطة XmlDocument ، إلخ)،يوجد أيضاً عدة طرق لتحويل البيانات.إذا كنت تريد فقط التنقل من ملف مدخلات input file إلى ملف مخرجات output file، فإن الكود التالي يوفر طريقة فعالة وسريعة. إنه يستخدم حالة System.Xml.Xsl.XslCompiledTransform لإتمام العمل.

```
' ----- Above: Imports System.Xml.Xsl
        Dim xslTrans As XslCompiledTransform
        ' ----- Open the XSL file as a transformation.
        xslTrans = New XslCompiledTransform()
        xslTrans.Load("c:\convert.xsl")
        ' ----- Convert and save the output.
        xslTrans.Transform("c:\input.xml", "c:\output.txt")
```

استخدام XML في الدوت نت:الطريقة الجديدة.Using XML in .NET: The New Way

عندما ظهرت الفيجوال بيسك للمرة الأولى،لم يكن أحد قد سمع بـ XML.ولكن الآن إنها في كل مكان.والآن في الفيجوال بيسك2008 أصبحت جزء من بناء اللغة نفسها.أين سينتهي بها المطاف؟

إن إخراج(إزالة) ما يجعل XML جزء من اللغة بعيد نوعاً ما.في مقطع الطريقة القديمة ،أريتك بعض الكود الذي ينشئ XMLلقائمة المنتج product list من أجل "Chai " محتوى XML كان بطول11سطر ،ولكنه يأخذ تقريباً 50 سطر من الكود المصدري لإنتاجه.ولكن تستطيع بناء نفس محتوى XML باستخدام الطريقة الجديدة بحوالي 11 سطر تقريباً.

```
Dim chaiItem As System.Xml.Linq.XDocument = <?xml version="1.0"?>
        <productList>
            <!-- We currently sell these items. -->
            <supplier ID="652" fullName="Beverages R Us">
                <product ID="1" available="Yes">
```

```
                    <productName>Chai</productName>
                    <category>Beverages</category>
                    <unitPrice>18.00</unitPrice>
                </product>
            </supplier>
        </productList>
```

ما عدا سطر التصريح الأول.المحتوى مطابق ل XML النهائي.إن ميزة XMLالحرفية الجديدة *XML Literals* feature new تجعل بناء مستندات XML موجز ومحكم.يتم تخزين المحتوى في كائن XDocumentجديد، جزء من فضاء الأسماء System.Xml.Linq. إذا أردت تخزين فقط مقطع XMLبدلاً من كامل المستند،استخدم الفئة XElementبدلاً عن ذلك.

```
Dim productSection As System.Xml.Linq.XElement = _
<product ID="1" available="Yes">
    <productName>Chai</productName>
    <category>Beverages</category>
    <unitPrice>18.00</unitPrice>
</product>
```

إذا كتبت تمكين الاستنتاج(الاستدلال) في برنامجك(خيار الاستنتاج فعالOption Infer On)،فلا تحتاج حتى لإخبار الفيجوال بيسك فيما إذا كان XElementأو XDocument.

```
  Dim productSection = _
<product ID="1" available="Yes">
    <productName>Chai</productName>
    <category>Beverages</category>
    <unitPrice>18.00</unitPrice>
</product>
```

فبمجرد تشابهه مع فئة XmlDocument،تتضمن الفئة XDocumentالطرق "تحميل Load"و"حفظ Save " لإدارة ملف معتمد على XML.

تعابير XMLالمدمجة .Embedded XML Expressions

تضمين XMLفي كودك المصدري مادة مدهشة،ولكن ستبقى مدهشة لو اشتريت دائماً "الشاي Chai " بـ " 18.00$"لكل وحدة فقط.محتوى XMLالحقيقي يأتي عادة من بيانات متغيرة.وعلى الرغم من أن XMLاسم "حرفيLiteral "،يمكن لحرفيات XMLأن تتضمن محتوى غير حرفي من خلال تعابير *XML* المدمجة *embedded XML expressions*. حيثما تريد إلى إضافة بيانات من متغير أو تعبير لنص ،تستخدم الرموز =%>و>% لتعويض بياناتك المخصصة.

```
Dim productName As String = "Chai"
        Dim productCategory As String = "Beverage"
        Dim productPrice As Decimal = 18@
        Dim isAvailable As Boolean = True
        Dim productSection = _
        <product ID=<%= productID %>
            available=<%= Format(isAvailable, "Yes/No") %>>
            <productName><%= productName %></productName>
            <category><%= productCategory %></category>
            <unitPrice><%= productPrice %></unitPrice>
        </product>
```

بالطبع ،من أجل إنتاج تصنيف المنتجات products كامل،سيكون عليك عمل الكثير من الكتابة.في الفصل 17،سأقدم بعض الطرق الإضافية لتضمين تعابير XMLمع كامل جداول البيانات.

الخاصيات المحورية لـ XML. XML Axis Properties

سابقاً في هذا الفصل، في مقطع "إيجاد الإبر وكومات القش Finding Needles and Haystacks"،بينت لك كيفية الوصول إلى مقاطع نوعية لمستندات XML القديمة النمط.تتضمن كائنات XML الحديثة النمط أيضاً طرق لعمل مسح والوصول إلى أقسام XMLالشجرية.

وهذه تدعى الخاصيات المحورية لـ*XML*، وهي تأتي محزمة في ثلاث تنوعات بنائية محببة:

محور العضو الابن *Child-member axis*

تستطيع الوصول إلى وسم فرعي(ابن)مباشرةً لأي XElement باستخدام اسم الابن كعضو لكائن الأب ،تغليف اسم الفرع في مجموعة أقواس محددة:

```
childElement = parentElement.<childName>
```

محور العضو المنحدر *Descendent-member axis*

تنوع عن بناء محور العضو الابن يتيح لك الوصول إلى الأعضاء المسماة عند أي عمق ضمن عنصر الأب.بدلاً من استخدام فقط نقطة مفردة(.) بين أسماء الأب والابن،استخدم ثلاث نقاط(.):

```
setOfElements = parentElement...<descendentName>
```

محور المواصفة. *Attribute axis*

الوصول لأي مواصفة لعنصر بمعاملة اسم المواصفة كاسم عضو،اسبق اسم المواصفة بالرمز @ .

```
attributeValue = parentElement.@attributeName
```

المقطع التالي من الكود يعمل مسح على قائمة المنتج الذي صممناه سابقاً في الفصل،يعرض رقم المعرف IDواسم كل منتج على لوحة console .

```
For Each oneProduct In allProducts...<product>
```

```
            Console.WriteLine(oneProduct.@ID & ": " & _
            oneProduct.<productName>.Value)
        Next oneProduct
```

يستخدم هذا الكود ثلاث أنماط محوريةaxis styles. تعمل الحلقة For Each...Nextمسح على مدخلات <product>المتطابقة باستخدام محور العضو المنحدر descendent-member axis،في كل عنصر منتج متطابق،يصل الكود مواصفة IDباستخدام محور المواصفةattribute axis، ويحصل على اسم المنتج باستخدام محور العضو الابنchild-member axis، على طول مع خاصية القيمة `Value` لعنصر الابن المعادة.تبدو المخرجات كالتالي:

```
1: Chai
2: Chang
3: Aniseed Syrup
```

فضاءات الأسماء والمخططات لمحارف XML. Namespaces and Schemas for XML Literals

كما مع الطريقة القديمة في إدارة XML،تسمح لك الطريقة الحديثة تضمين فضاءات أسماء في محتوى XML.لإضافة سابقة فضاء أسماءXML، ببساطة ضمنه في المحتوى كما تعمل مع أي سيناريو أخر.

```
Dim foodItems = _
 <?xml version="1.0"?>
 <menu:items xmlns:menu="http://www.timaki.com/menu">
     <menu:item>
         <menu:name>Pizza</menu:name>
         <menu:price>12.99</menu:price>
     </menu:item>
     <menu:item>
         <menu:name>Breadsticks</menu:name>
         <menu:price>1.99</menu:price>
     </menu:item>
 </menu:items>
```

تستطيع أيضاً تعريف فضاء الأسماء،الجزءxmlns، باستخدام تباين(تنوع)variation) عبارة الفيجوال بيسك Imports .

```
Imports <xmlns:menu="http://www.timaki.com/menu">
    '...later...
Dim foodItems = _
<?xml version="1.0"?>
<menu:items>
...and so on...
```

ستحافظ الفيجوال بيسك على إدراج تعريف xmlnsفي المكان المناسب في محتوىXML. يتم تخزينه عملياً ككائن XNamespaceمتميز ضمن XDocument أو XElement.لإنتاج كائن XNamespaceما لاستخدامك الخاص،تتضمن الفيجوال بيسك الوظيفة (الدالة) GetXmlNamespaceالجديدة.

```
Dim justTheNamespace = GetXmlNamespace(menu)
```

مشروع. Project

يريد مدير نظام المكتبة رؤية الإحصائيات والمعلومات بلمحة سريعة،أو تشغيل التقارير المتنوعة التي توفر ملخص هام أو عرض تفاصيل بيانات النظام.على الرغم من أنني وكمبرمج أستطيع تجريب إضافة كل نوع محتمل الحدوث conceivableمن التقارير التي يحتاجها المستخدم،فقد تعلمت من خبرتي أن هذا غير ممكن.فدائماً يريد المستخدم القمر(الزبدة)،عادةً في نموذج مقتصر على تقرير والذي وعلى حد علمي سيتم استخدامه مرة واحدة وبعدها لن يُنظر إليه مرةً أخرى. إني لا أحب إعادة تجميع وتحرير كامل التطبيق كلما أراد مستخدم تقرير جديد.بدلاً عن ذلك،إني احفظ التقارير خارج نطاق التطبيق،أخزنها كبرامج منفصلة. ومن ثم،من نموذج واحد في التطبيق الرئيسي،أجعل جميع هذه التقارير الخارجية متاحة في قائمة مناسبة مريحة.
لتنفيذ هذه الميزة الشاملة ،استخدم ملف تركيب تقرير،ملف XMLبسيط يحتوي على معلومات عن التقارير المتاحة،وكيفية تشغيلها.أريد قائمة الاختيار الخاصة بي أن تتضمن بنود فارغة بحيث أستطيع تجميع التقارير بشكل مرئي وبوسائل مريحة.
لعمل هذا ،سأعمل ملف XMLضمن تسلسل هرمي بعمق غير محدد،مع تمثيل كل مستوى بأبعد مستوى من الفراغ indent المعروض.على سبيل المثال،لنقول أنني أريد الخطوط الرئيسية التالية من التقارير(مع عناوين مجموعة التقرير بخط غامق).

تقارير التفاصيلDetail Reports
التقارير اليوميةDaily Report
التقارير الشهريةMonthly Reports
القيم الشهريةMonthly Value
الجرد الشهري Monthly Inventory
تقارير الإيجاز(التلخيص)Summary Reports
ملخص الجردInventory Summary

إعداد XML سيتبع التركيب التالي:

```
<Group name="Detail Reports">
  <Item name="Daily Report"/>
  <Group name="Monthly Reports">
    <Item name="Monthly Value"/>
    <Item name="Monthly Inventory"/>
  </Group>
</Group>
```

```
<Group name="SummaryReports">
  <Item name="Inventory Summary"/>
</Group>
```

بالطبع هذا XML بسيط جداً(بغض النظر عن كونه غير ميسر noncompliant)بالإضافة إلى الهرمية،أريد أيضاً تضمين دعم لطرق التقرير المتنوعة. لحفظ الأشياء بسيطة،سيتضمن مشروع المكتبة ثلاث أنواع من التقارير:

التقارير الجاهزة(المبنية داخلياً).Built-in reports

يتضمن التطبيق عدد محدد من التقارير التي تبنى بشكل دائم في التطبيق الرئيسي(المجمع assembly).تم ترقيم التقارير بدءاً من 1،وعند هذه المرحلة لدي خمسة تقارير في مخيلتي.يمكن أن يختار مصمم ملف إعداد XML تضمينها في عرض التقارير أو لا، بكل بساطة بتضمينها أو عدم تضمينها في الملف.في حال غياب absence ملف الإعداد، ستظهر هذه التقارير في القائمة بشكل افتراضي.بالإضافة إلى عدد التقارير (من 1 إلى5)،كل مدخلة لديها نص عرض display text ووصف طويل long description .

تقارير التطبيق.Application reports

تكون هذه التقارير ملفات تنفيذية EXE متميزة ومفصولة،وتبدأ بواسطة طرق تمهيد initiation التطبيق القياسية.كل مدخلة تتضمن نص العرض display text،المسار الكامل للتطبيق،معاملات نسبية اختيارية،علامة لتمرير هوية المستخدم identity of the user التي تمهد(تسند قيمة أولية)للتقرير،وشرح طويل.

تقارير عناوين الانترنت: URL reports

هذه التقارير هي استدعاء بسيط لصفحات ويب،أو أي معرف انترنت URL. كل مدخلة تتضمن نص العرض،معرف الانترنت URL نفسه،ووصف طويل.

نشاطات المشروع في هذا الفصل تتضمن كل من كتابة الكود وتوثيق مورد خارجي جديد(تنسيق ملف XML)

تحديث التوثيق التقني.Update Technical Documentation

أولاً،دعنا نضيف توثيق واضح على تركيب ملف هيئة XML. لا توجد طريقة سهلة لوصل تركيب ملف XML إلى مستخدم اعتيادي،على الرغم من أن مثل هذا التوثيق مطلوب،على أمل أن يتضمن التطبيق أيضاً أداة تتيح للمدير بناء ملف التركيب(الإعداد).

ملف تركيب التقرير.Report Configuration File

يمكن أن يتم إعداد تطبيق المكتبة ليشغل أي عدد من التقارير من خلال نموذج التقارير.يتم إدارة قائمة التقارير المتاحة من خلال ملف تركيب تقرير XML.

يحتوي الملف "مجموعات groups " و"بنود items ".جميع البنود هي تقارير،وتظهر ضمن مجموعة.تستطيع إدخال مجموعة ضمن مجموعة أخرى لأي عمق،وقائمة التقارير المعروضة في برنامج المكتبة ستترك فراغ لكل مجموعة تابعة(ثانوية)لمساعدة المستخدم على رؤية تنظيم التقارير.لايوجد حد لتداخل المجموعات.

العنصر الجذري لملف XML يجب أن يتم تسميته<reportList> ،ويمكن أن يحتوي على أي عدد من عناصر البيانات<reportGroup> و<reportItem>:

<reportItem>:يمثل مدخلة تقرير مفرد.وهذه المدخلة لديها مواصفة attribute واحدة مطلوبة ،وتصل إلى خمسة عناصر بيانات تابعة(ثانوية subordinate)

وتعتمد على إعدادات المواصفة:

ـــ نوع type (المواصفة attribute):تعمل الإعداد إلى واحد من القيم التالية:

. جاهز(مبني داخلياً) built-in:تشغيل واحد من البرامج الجاهزة(المبنية داخلياً).هذا النوع من التقارير يستخدم عناصر البيانات <displayText>، <reportPath> و<description> .

.برنامج program:تشغيل برنامج تنفيذي EXE منفصل.هذا النوع من التقارير يستخدم عناصر البيانات<displayText>، <reportPath>، <reportArgs>، <reportFlags>، و<description> .

.معرف ويب url :يشغل URL (محدد مواضع الموارد الموحد ***Uniform Resource Locator***)،مثل صفحة ويب أو"بروتوكول البريد mailto " ايميل إلى عنوان مستقبل.هذا النوع من التقارير يستخدم عناصر البيانات<displayText>، <reportPath> و<description>.

ـــ عرض نص<displayText> :اسم مختصر أو وصف لهذا التقرير،كما سيظهر في قائمة خيارات التقرير.هذا العنصر مطلوب لجميع أنواع التقارير.

ـــ مسار التقرير<reportPath> :المسار الكامل full path،محدد مواضع الموارد الموحد URL،أو عدد التقارير number of the report،بالاعتماد على نوع التقرير.من أجل تقارير برنامج(تنفيذي EXE)،هذا هو المسار UNC الكامل أو مسار المعتمد على حرف السواقة للتقرير،بدون معاملات نسبية إضافية.من أجل التقارير الجاهزة(المبنية داخلياًbuilt-in)هذا هو رقم التقرير،من 1إلى 5(القيم ومعانيها تم جدولتها فيما بعد في هذا الفصل).من أجل تقارير URL،هذا هو محدد المواضع الفعلي،كما في "*http://mysite.com/myreport.aspx*" أو "*mailto:helpdesk@mysite.com*".وهذا العنصر مطلوب من أجل جميع أنواع التقارير.

ـــ معاملات التقرير<reportArgs> :من أجل تقارير برنامج (تنفيذي EXE)،هذه المدخلة أي معاملات نسبية arguments لسطر أمر commandline يمكن تضمينه عند تشغيل البرنامج.هذا العنصر محقق فقط في تقارير البرنامج(التنفيذي)،وهو دائماً اختياري.

إشعارات التقرير <reportFlags>:من أجل تقارير برنامج(تنفيذي EXE)،هذه المدخلة هي الدلالات الاختيارية التي يجب إلحاقها إلى أمر التطبيق كمعاملات نسبية.عند هذه المرحلة الإشعار الوحيد هو U.عندما يتم وضع هذا العنصر إلى U،يتم إلحاق المعامل النسبي*userid* -u إلى نص الأمر(حيث *userid* هو معرف تسجيل دخول المستخدم من حقل قاعدة البيانات UserName.LoginID).هذا العنصر متاح فقط في تقارير البرنامج التنفيذي EXE ،وهو دائما اختياري.

ـــ الوصف<description> :هذا هو الأطول،وصف كثير الحشو verbose للتقرير،يصل إلى 200حرف،والتي ستظهر على نموذج التقرير Report form عندما يختار المستخدم قائمة نموذج التقرير. سيساعد هذا الشرح المستخدم في اختيار التقرير المناسب.هذا العنصر متاح لجميع أنواع التقارير،ولكن دائماً اختياري.

. مجموعة التقرير<reportGroup>:تمثل مجموعة التصنيف category group ،تستخدم للمجموعة المرئية وترك فراغ في بداية indent التقارير في قائمة العرض.يجب أن يحتوي هذا العنصر بالضبط العنصر<displayText> الوحيد ،ولكن يمكن أن يحتوي أي عدد من العناصر<reportItem> أو<reportGroup>

ـــ <displayText>نص العرض :اسم مختصر أو وصف لهذه المجموعة،كما ستظهر في قائمة اختيار تقرير.هذا العنصر مطلوب.

عند استخدام نوع التقرير"الجاهز" يتم وضع العنصر إلى واحد من قيم الانتغر التالية:

1.ـــ تقرير مخرجات(مراجعة) البنود.Items Checked Out Report

2.ـــ تقرير متأخرات للبنود.Items Overdue Report

3.ـــ تقرير فقدان بنود.Items Missing Report

4.—— مستحقات الغرامات بواسطة تقرير الزبائن.Fines Owed by Patrons Report

5.—— تقرير إحصاءات قاعدة بيانات المكتبة.Library Database Statistics Report

هذا الوصف التقني يظهر في مستند مجموعة المصدر التقني،تم تطويره أصلاً في الفصل الرابع.

إنشاء فئة مدخلة تقرير. Create Report Entry Class

مع إمكانيات الدوت نت لتخزين جميع الكائنات كبنود صندوق قائمةListBox items ،نستطيع إنشاء فئة خاصة بحيث تحتوي جميع المعلومات المطلوبة لاختيار وتشغيل تقرير من قائمة التقارير. هذه الفئة بسيطة جداً،لايوجد فيها شيء ماعدا حقول عامة أساسية،زائد إعادة قيادة لدالة " ToString"،والمستخدمة بواسطة أداة صندوق القائمة لعرض كل بند قائمة بشكل مناسب.

في مشروع المكتبة،أضف ملف فئة جديدة وسمه ReportItem.vb من خلال مشروع Project>> إضافة فئةAdd Class. ومن ثم أضف العداد التالي لهذا الملف،ولكن أضفه خارج الحدود Class...End Class. يشير هذا العداد إلى ما يمثله نوع مدخلة كل بند قائمة.كما يلي:

```
Public Class ReportItem
.
.
.
End Class
Public Enum ReportItemEnum
'نوع بند في قائمة اختيار تقرير
    GroupLabel = 0
    BuiltInCheckedOut = 1
    BuiltInOverdue = 2
    BuiltInMissing = 3
    BuiltInFinesOwed = 4
    BuiltInStatistics = 5
    ExeProgram = 6
    UrlProgram = 7
End Enum
```

أضف إلى نفس الملف، أعضاء الفئة ReportItem.تحتوي هذه الفئة جميع المعلومات التي نحتاجها لتشغيل التقارير المحملة من ملف التركيب configuration file.

```
'نسخة من بنود اختيار تقرير المستخدمة في نموذج
    'ReportSelect
    Public ItemType As ReportItemEnum
'مستوى الفراغ يبدأ بصفر
    Public Indent As Integer
    Public DisplayText As String
    Public ReportPath As String  ' ExeProgram / UrlProgram only
    Public ReportArgs As String  ' ExeProgram only
    Public Description As String
    Public Overrides Function ToString() As String
        ' بفراغ مسبوق نص عرض.
        Return StrDup(Indent * 5, " ") & DisplayText
    End Function
```

تصميم نموذج التقارير.Design the Report Form

يستخدم أمناء المكتبة Librariansوالمدراء administratorsنموذج اختيار تقرير(شاهد الشكل التالي) لعرض التقارير. يتضمن النموذج أداة صندوق قائمة ListBoxتعرض جميع التقارير ومجموعات التقارير، زر التشغيل Runالذي يشغل التقرير،وزر الإغلاق الذي يعود بالمستخدم إلى النموذج الرئيسي.تعرض أداة عنونة label وصف كامل عن التقرير،عندما يكون متاح، تحت صندوق القائمة تماماً.



لإضافة هذا النموذج إلى مشروعك(راجع مشروع هذا الفصل).

على الرغم من أن المدير عمل على منح أسماء مفيدة لكل تقرير،الاسم المختصر لاسم كل تقرير من المحتمل أنه مايزال غامض بالنسبة للمستخدم.يتضمن كل تقرير وصف كامل اختياري.عندما يختار المستخدم قائمة نموذج التقارير،يعمل معالج حدث على تحديث اللافتة label " FullDescription "التي تحت القائمة تماماً.أضف معالج الحدث هذا.

```
Private Sub AllReports_SelectedIndexChanged(ByVal sender As Object, ByVal e As System.EventArgs)
Handles AllReports.SelectedIndexChanged
' عرض وصف التقرير,إذا كان متاح
        Dim reportEntry As Library.ReportItem
' إزالة أي وصف سابق
        FullDescription.Text = "ختار تقرير لايوجد"
        If (AllReports.SelectedIndex <> -1) Then
' إيجاد المحتوى وعرضه
            reportEntry = CType(AllReports.SelectedItem, Library.ReportItem)
            FullDescription.Text = reportEntry.Description
        End If
    End Sub
```

Populate Reports from Configuration File.ملئ التقارير من ملف التركيب

تحمل الطريقة RefreshReportListالبيانات من ملف تركيب التقرير وتعالج النتائج.في النهاية،سيتم تسجيل موقع هذا الملف في ملف التركيب للتطبيق،ولكن لن نعمل على إضافته حتى فصل لاحق.

أما الآن،دعنا ندخل موقع ملف الاختبار،ونعلّمه من أجل التحديث اللاحق.فضلت اختيار كائنات XMLللنمط القديم من أجل هذا الكود لأن ميزات XMLللنمط الجديد التي نحتاجها لجعل الكود سهل الكتابة لن يتم تقديمها حتى فصل لاحق.

```
Private Sub RefreshReportList()
' التحميل بالقائمة التقارير المتاحة
        Dim configFile As String
        Dim configData As Xml.XmlDocument
        Dim reportEntry As ReportItem
        Dim counter As Integer
        On Error GoTo ErrorHandler
' تنظيف القائمة الموجودة
        AllReports.Items.Clear()
' الحصول على موقع ملف التركيب
' لعمل هذا:حمل هذا من تركيب التطبيق,حالياً علينا فقط كتابة كود القيمة
        configFile = "c:\ReportConfig.txt"
' تحميل ملف التركيب
        If (configFile <> "") Then
            If (System.IO.File.Exists(configFile)) Then
' تحميل الملف
                configData = New Xml.XmlDocument
                configData.Load(configFile)
' معالج ملف التركيب
                LoadReportGroup(configData.DocumentElement, 0)
            End If
        End If
' إذا كان لم ينتج عن ملف التركيب أي تقرير ظاهر في القائمة
' أضف التقارير الافتراضية
        If (AllReports.Items.Count = 0) Then
            For counter = 1 To CInt(ReportItemEnum.BuiltInStatistics)
' بناء مدخلة التقرير
                reportEntry = New ReportItem
                reportEntry.Indent = 0
                reportEntry.ItemType = CType(counter, ReportItemEnum)
                Select Case reportEntry.ItemType
                    Case ReportItemEnum.BuiltInCheckedOut
                        reportEntry.DisplayText = "إخراجها تم أو معارة البنود"
                    Case ReportItemEnum.BuiltInOverdue
                        reportEntry.DisplayText = "متأخرة البنود"
                    Case ReportItemEnum.BuiltInMissing
                        reportEntry.DisplayText = "مفقودة البنود"
                    Case ReportItemEnum.BuiltInFinesOwed
                        reportEntry.DisplayText = "بغرامات مدين الزبون"
                    Case ReportItemEnum.BuiltInStatistics
                        reportEntry.DisplayText = "البيانات قاعدة إحصاءات"
                End Select
' أضف مدخلة التقرير إلى القائمة
                ' ----- Add the report entry to the list.
                AllReports.Items.Add(reportEntry)
            Next counter
```

```
        End If
        Return
ErrorHandler:
        GeneralError("ReportSelect.RefreshReportList", Err.GetException())
        Resume Next
    End Sub
```

لأن ملف تركيب التقرير يسمح بتداخل مجموعات التقرير لأي مستوى،نحتاج إلى استخدام روتين دوري recursiveللنزول إلى كل مستوى متتابع(متتالي) مرةً بعد مرة.يتم استدعاء الروتين LoadReportGroupبواسطة الروتينRefreshReportList ،ويضيف جميع بنود التقارير ومجموعات التقارير ضمن مجموعة بدء التقرير.يتم استدعاءه بشكل أولي(LoadReportGroup) من نقطة مرجعية للعنصر الجذري<reportList> .كل مرة يعمل على إيجاد العنصر الابن<reportGroup>،يستدعي نفسه مرةً أخرى ولكن هذه المرة يبدأ من نقطة مرجعية reference point للعنصر الابن<reportGroup>.

```
Private Sub LoadReportGroup(ByVal groupNode As Xml.XmlNode, ByVal indentLevel As Integer)
        ' أضف مجموعات وبنود عند هذا المستوى.
        'استدعي و ما تحتاج.
        Dim scanNode As Xml.XmlNode
        Dim detailNode As Xml.XmlNode
        Dim reportEntry As ReportItem
        On Error GoTo ErrorHandler
        ' عالج كل بند أو مجموعة.
        For Each scanNode In groupNode.ChildNodes
            'بناء بند لتضمينه في القائمة.
            reportEntry = New ReportItem
            reportEntry.Indent = indentLevel
            ' احصل على الاسم المعروض.
            detailNode = scanNode.SelectSingleNode("displayText")
            If (detailNode Is Nothing) Then Continue For
            reportEntry.DisplayText = Trim(detailNode.InnerText)
            If (scanNode.Name = "reportGroup") Then
                ' ابدأ عرض مجموعة جديدة.
                reportEntry.ItemType = ReportItemEnum.GroupLabel
                AllReports.Items.Add(reportEntry)
                'عد إلى البنود الأبناء
                LoadReportGroup(scanNode, indentLevel + 1)
            ElseIf (scanNode.Name = "reportItem") Then
                ' هذا بند, سجل موقعه.
                detailNode = scanNode.SelectSingleNode("reportPath")
                If Not (detailNode Is Nothing) Then reportEntry.ReportPath =
Trim(detailNode.InnerText)
                ' احصل على أي معامل نسبي لسطر أمر.
                detailNode = scanNode.SelectSingleNode("reportArgs")
                If Not (detailNode Is Nothing) Then reportEntry.ReportArgs =
Trim(detailNode.InnerText)
                ' احصل على اية علامات بند معين
                detailNode = scanNode.SelectSingleNode("reportFlags")
                If Not (detailNode Is Nothing) Then
                    If (InStr(UCase(detailNode.InnerText), "U") > 0) And (LoggedInUserName <> "")
Then reportEntry.ReportArgs = _
                        Trim(reportEntry.ReportArgs & " -u " & LoggedInUserName)
                End If
                detailNode = scanNode.SelectSingleNode("description")
                If Not (detailNode Is Nothing) Then reportEntry.Description =
Trim(detailNode.InnerText)
                If (scanNode.Attributes("type").Value = "built-in") Then
                    If (IsNumeric(reportEntry.ReportPath) = False) Or (Val(reportEntry.ReportPath)
< 1) Or (Val(reportEntry.ReportPath) > _
                        CInt(ReportItemEnum.BuiltInStatistics)) Then Continue For
                    reportEntry.ItemType = CType(CInt(reportEntry.ReportPath), ReportItemEnum)
                    AllReports.Items.Add(reportEntry)
                ElseIf (scanNode.Attributes("type").Value = "program") Then
                    If (reportEntry.ReportPath = "") Then Continue For
                    reportEntry.ItemType = ReportItemEnum.ExeProgram
                    AllReports.Items.Add(reportEntry)
                ElseIf (scanNode.Attributes("type").Value = "url") Then
                    If (reportEntry.ReportPath = "") Then Continue For
                    reportEntry.ItemType = ReportItemEnum.UrlProgram
                    AllReports.Items.Add(reportEntry)
                End If
            End If
        Next scanNode
```

```
        Return
ErrorHandler:
        GeneralError("ReportSelect.LoadReportGroup", Err.GetException())
        Resume Next
    End Sub
```

اعمل على إضافة حدث تحميل Loadالفورم،والذي يحمل في محتوى الفورم ملف التركيب.

```
Private Sub ReportSelect_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
        RefreshReportList()
    End Sub
```

Running the Reports.تشغيل التقارير

والآن ،هذه جميع المجموعات والتقارير التي تظهر في القائمة،علينا تشغيل التقارير الفعلية.معالج حدث زر "تشغيل ActRun " يعالج هذه المهمة.حالياً سنعمل على إضافة فقط إطار العمل لدعم الاستدعاء بالنسبة لكل تقرير.أما تقارير"المبنية داخلياً built-in (الجاهزة)"سيتم إضافتها في الفصل21.

```
  Private Sub ActRun_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
ActRun.Click
تشغيل التقرير المختار        '
        Dim reportEntry As Library.ReportItem
        On Error GoTo ErrorHandler
التأكد من أن التقرير قد تم اختياره        '
        If (AllReports.SelectedIndex = -1) Then
            MsgBox("من فضلك اختر تقرير من القائمة", MsgBoxStyle.OkOnly Or MsgBoxStyle.Exclamation, _
               ProgramTitle)
            Return
        End If
كود مختلف من أجل كل نوع مدخلة        '
        reportEntry = CType(AllReports.SelectedItem, Library.ReportItem)
        Me.Cursor = Windows.Forms.Cursors.WaitCursor
        Select Case reportEntry.ItemType
            Case ReportItemEnum.GroupLabel
لاتقرير من أجل مدخلات مجموعة        '
                MsgBox("من فضلك اختر تقرير من القائمة", MsgBoxStyle.OkOnly Or
MsgBoxStyle.Exclamation, ProgramTitle)
            Case ReportItemEnum.BuiltInCheckedOut
البنود تم استعارتها أو اخراجها        '
 للعمل:اكتب        '
                'BasicReportCheckedOut()
            Case ReportItemEnum.BuiltInOverdue
البنود متأخرة        '
للعمل:اكتب        '
                ' BasicReportOverdue()
            Case ReportItemEnum.BuiltInMissing
البنود مفقودة        '
 للعمل: اكتب        '
                ' BasicReportMissing()
            Case ReportItemEnum.BuiltInFinesOwed
الغرامات المستحقة على الزبون        '
 للعمل:اكتب        '
                ' BasicReportFines()
            Case ReportItemEnum.BuiltInStatistics
إحصاءات قاعدة بيانات المكتبة        '
للعمل:اكتب التقرير        '
                ' BasicReportStatistics()
            Case ReportItemEnum.ExeProgram
بدء البرنامج        '
                Process.Start("""" & reportEntry.ReportPath & """ " & _
                    reportEntry.ReportArgs)
            Case ReportItemEnum.UrlProgram
بدء عناوين الانترنت        '
                ' ----- Start a URL.
                Process.Start(reportEntry.ReportPath)
        End Select
        Me.Cursor = Windows.Forms.Cursors.Default
        Return
ErrorHandler:
        GeneralError("ReportSelect.ActRun_Click", Err.GetException())
        Resume Next
    End Sub
```

من أجل التقارير الخارجية،يستدعي معالج الحدث الطريقةProcess.Start .وهي طريقة مدهشة تقبل إما تعبير سطر أمر قياسي،أو عنوان انترنت URL محقق،أو عنوان صفحة ويب.

الاتصال بنموذج اختيار تقرير.Connecting the Select Report Form

لجعل التقارير متاحة بالنسبة للمستخدم،يجب علينا تمكين وصلة لنموذج التقرير من الفورم الرئيسي.فيما سبق عملنا على تضمين لوحة متميزة على تلك الفورم من أجل طباعة التقارير فقط.الزر"عمل التقاريرActDoReports "على تلك الفورم يعمل على إطلاق استدعاء هذا النموذج.أنشئ معالج حدث جديد للزر" عمل التقاريرActDoReports " وأضف الكود.

```
Private Sub ActDoReports_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles
ActDoReports.Click
        ReportSelect.ShowDialog()
    End Sub
```

والآن،لدينا فهم راسخ(عميق)لعالم XML،سندع الفيجوال بيسك تعمل كل أعمال معالجة العمل الصعب من أجل عمليات إعداد التطبيق.